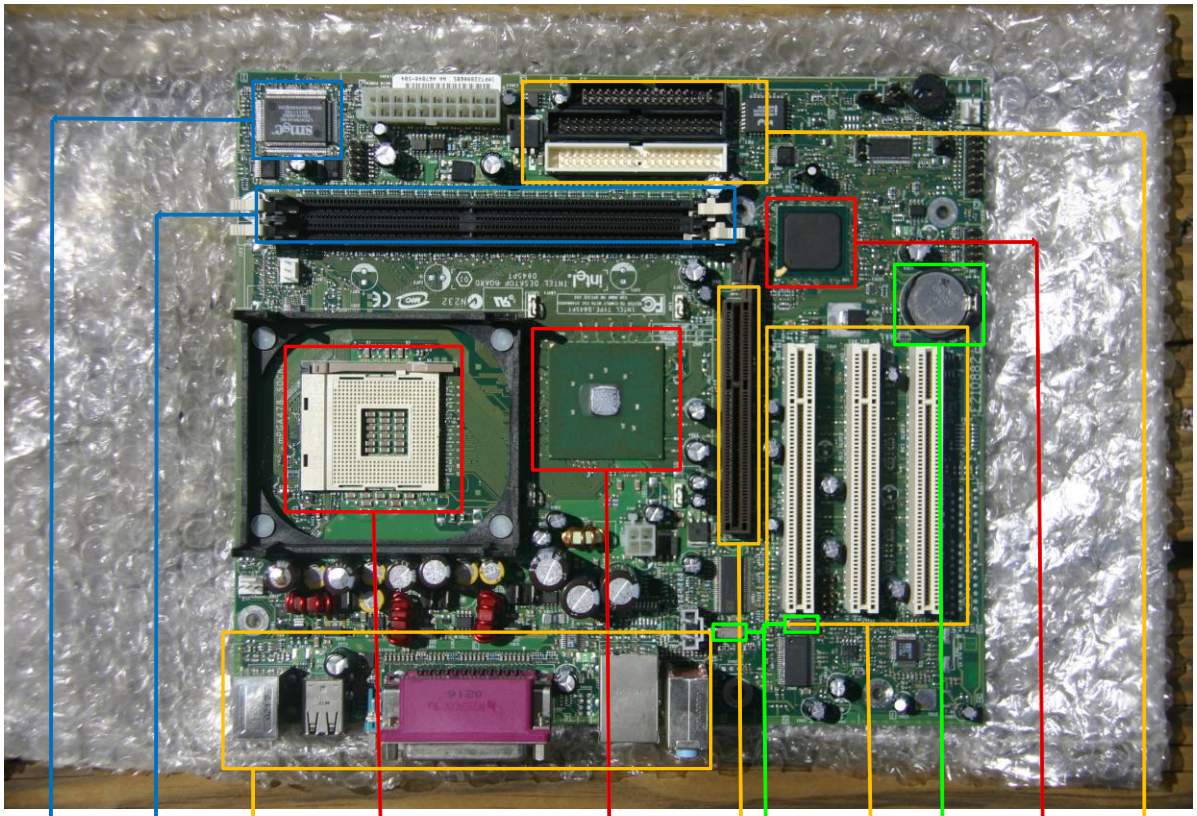
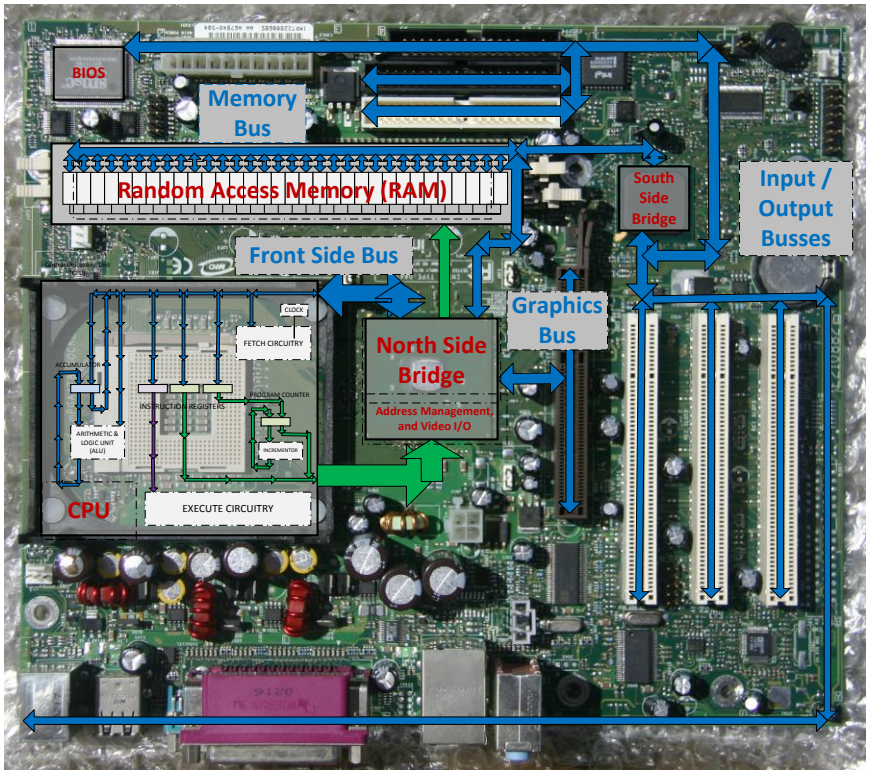
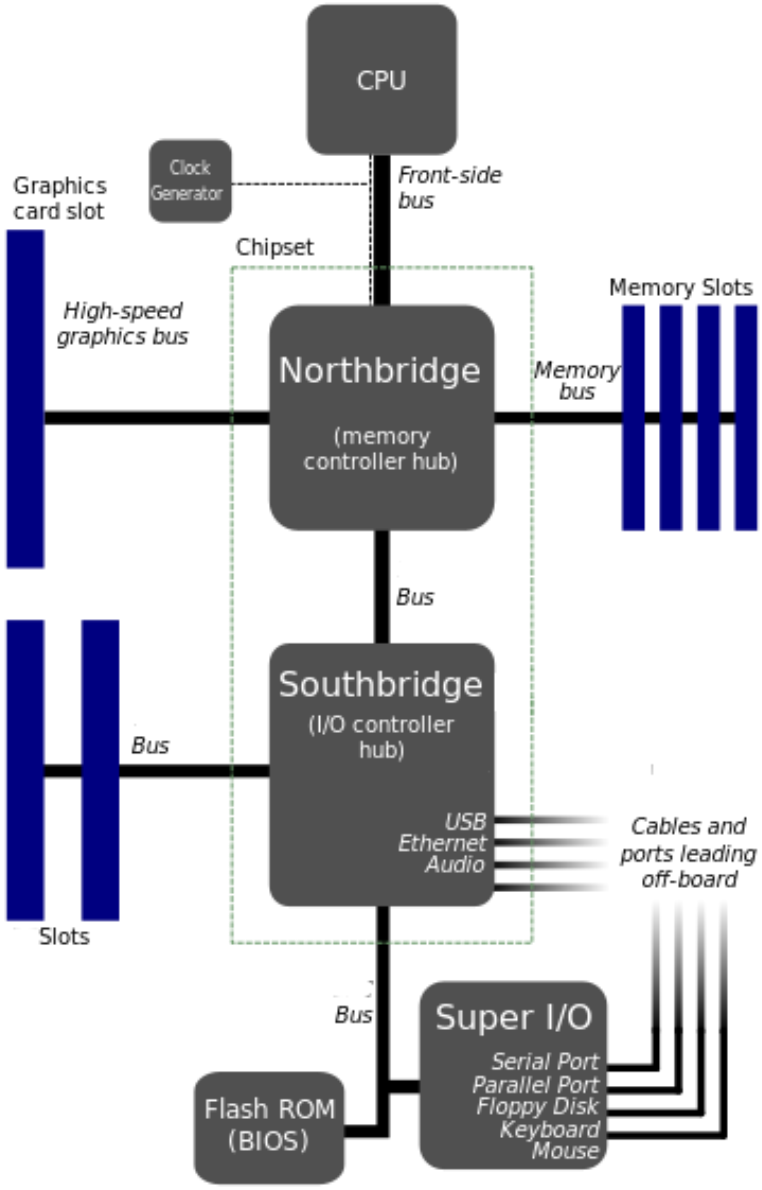
Central Processing Unit (CPU)

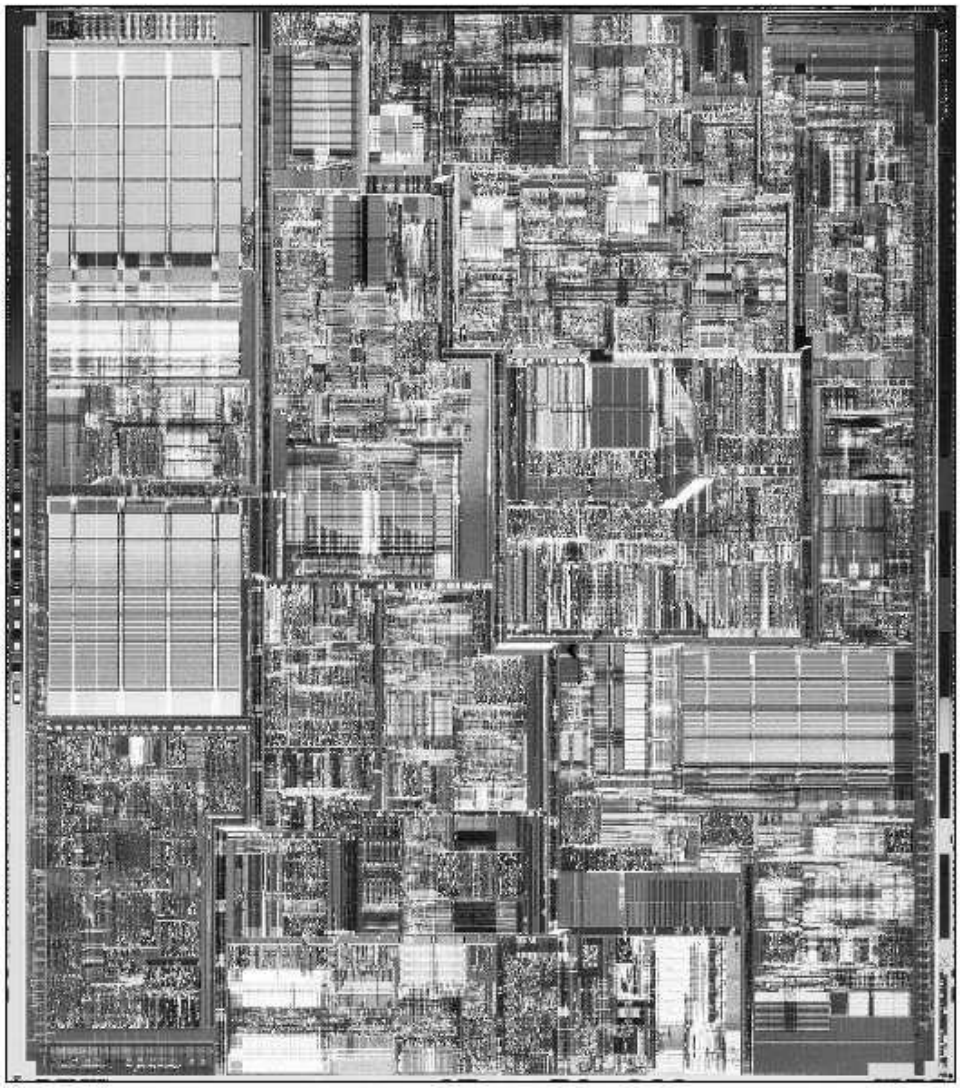
Bare Motherboard

all detachable components removed

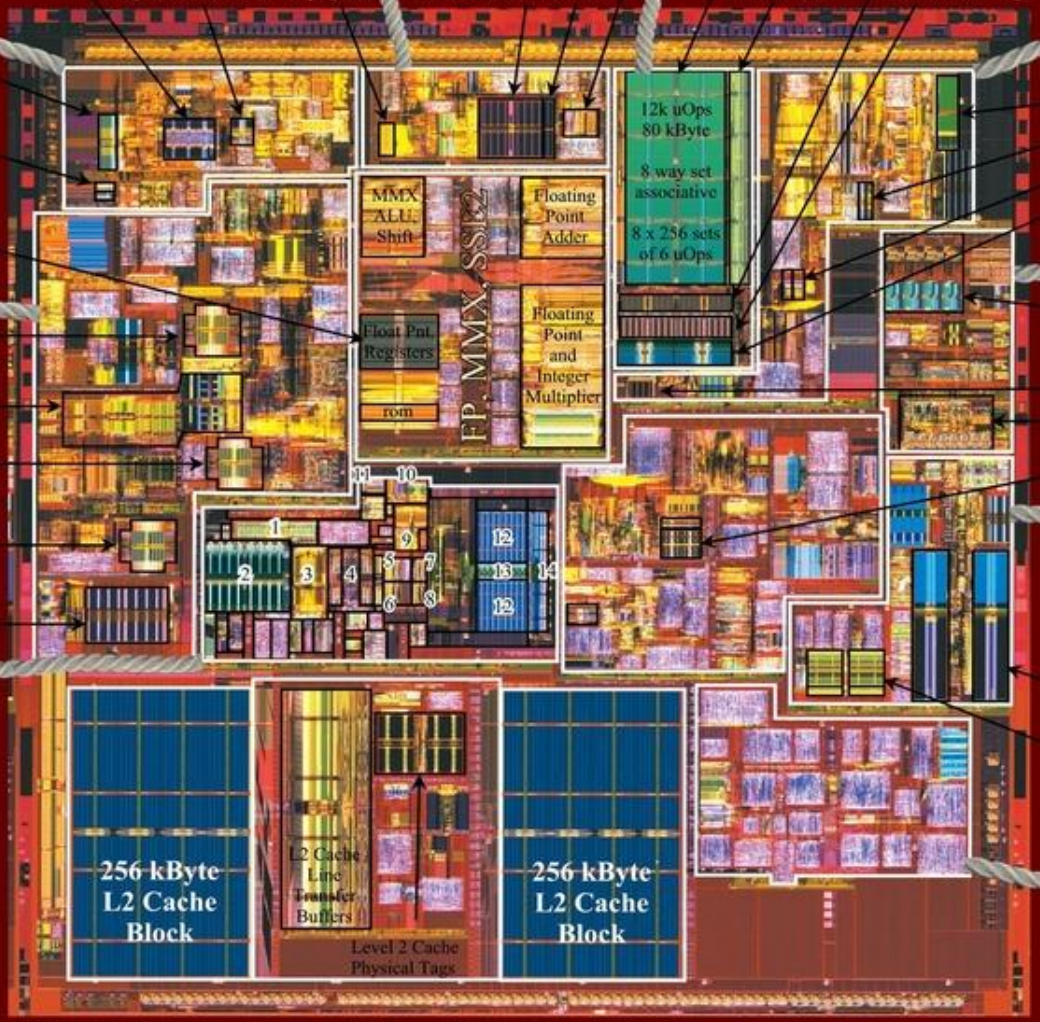

BIOS
Basic Input Output System Program
Memory
Random Access Memory (RAM)
I/O
External Sockets
Central Processing Unit (CPU)
socket only
North Bridge Chip
without heat sink
Video Card
Clock Crystals
Expansion Bus Sockets
Configuration Battery
for when power is shut off
South Bridge Chip
Disk I/O


BIOS
Memory Bus
Random Access Memory (RAM)
South Side Bridge
Input / Output Busses
Front Side Bus
Graphics Bus
North Side Bridge
Address Management, and Video I/O
CLOCK
FETCH CIRCUITRY
ACCUMULATOR
PROGRAM COUNTER
INSTRUCTION REGISTER
ARITHMETIC & LOGIC UNIT (ALU)
INCREMENTOR
CPU
EXECUTE CIRCUITRY


CPU
Clock Generator
Front-side bus
Graphics card slot
Chipset
Memory Slots
High-speed graphics bus
Northbridge
(memory controller hub)
Memory bus
Bus
Southbridge
(I/O controller hub)
Bus
USB
Ethernet
Audio
Cables and ports leading off-board
Slots
Bus
Super I/O
Serial Port
Parallel Port
Floppy Disk
Keyboard
Mouse
Flash ROM (BIOS)



Intel Pentium 4 Northwood



Buffer Allocation & Register Rename

Instruction Queue (for less critical fields of the uOps)

General Instruction Address Queue & Memory Instruction Address Queue (queues register entries and latency fields of the uOps for scheduling)

Floating Point, MMX, SSE2 Renamed Register File 128 entries of 128 bit.

uOp Schedulers

FP Move Scheduler: (8x8 dependency matrix)

Parallel (Matrix) Scheduler for the two double pumped ALU's

General Floating Point and Slow Integer Scheduler: (8x8 dependency matrix)

Load / Store uOp Scheduler: (8x8 dependency matrix)

Load / Store Linear Address Collision History Table

Integer Execution Core

(1) uOp Dispatch unit & Replay Buffer Dispatches up to 6 uOps / cycle
(2) Integer Renamed Register File 128 entries of 32 bit + 6 status flags 12 read ports and six write ports
(3) Databus switch & Bypasses to and from the Integer Register File.
(4) Flags, Write Back
(5) Double Pumped ALU 0
(6) Double Pumped ALU 1
(7) Load Address Generator Unit
(8) Store Address Generator Unit
(9) Load Buffer (48 entries)
(10) Store Buffer (24 entries)

Execution Pipeline Start

Instruction Trace Cache

Trace Cache Access, next Address Predict

Trace Cache Branch Prediction Table (BTB), 512 entries.

Return Stacks (2x16 entries)

Trace Cache next lP's (2x)

Miscellaneous Tag Data

Instruction Decoder

Up to 4 decoded uOps/cycle out. (from max. one x86 instr/cycle) Instructions with more than four are handled by Micro Sequencer

Trace Cache LRU bits

Raw Instruction Bytes in

Data TLB, 64 entry fully associative, between threads dual ported (for loads and stores)

Instruction Fetch from L2 cache and Branch Prediction

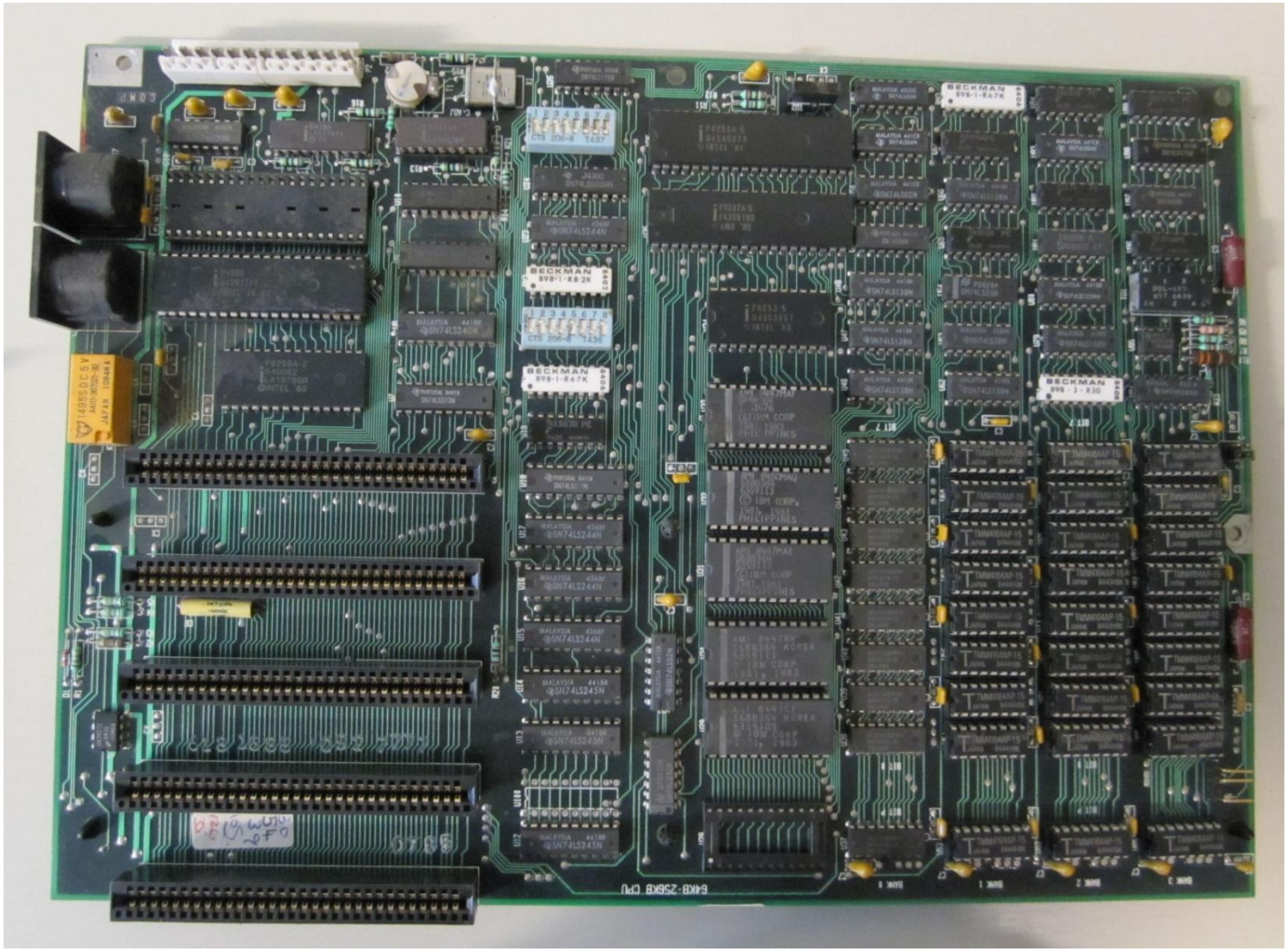
Front End Branch Prediction Tables (BTB), shared, 4096 entries in total

Instruction TLB's 2x64 entry, fully associative for 4k and 4M pages. In: Virtual address [31:12] Out: Physical address [35:12] + 2 page level bits

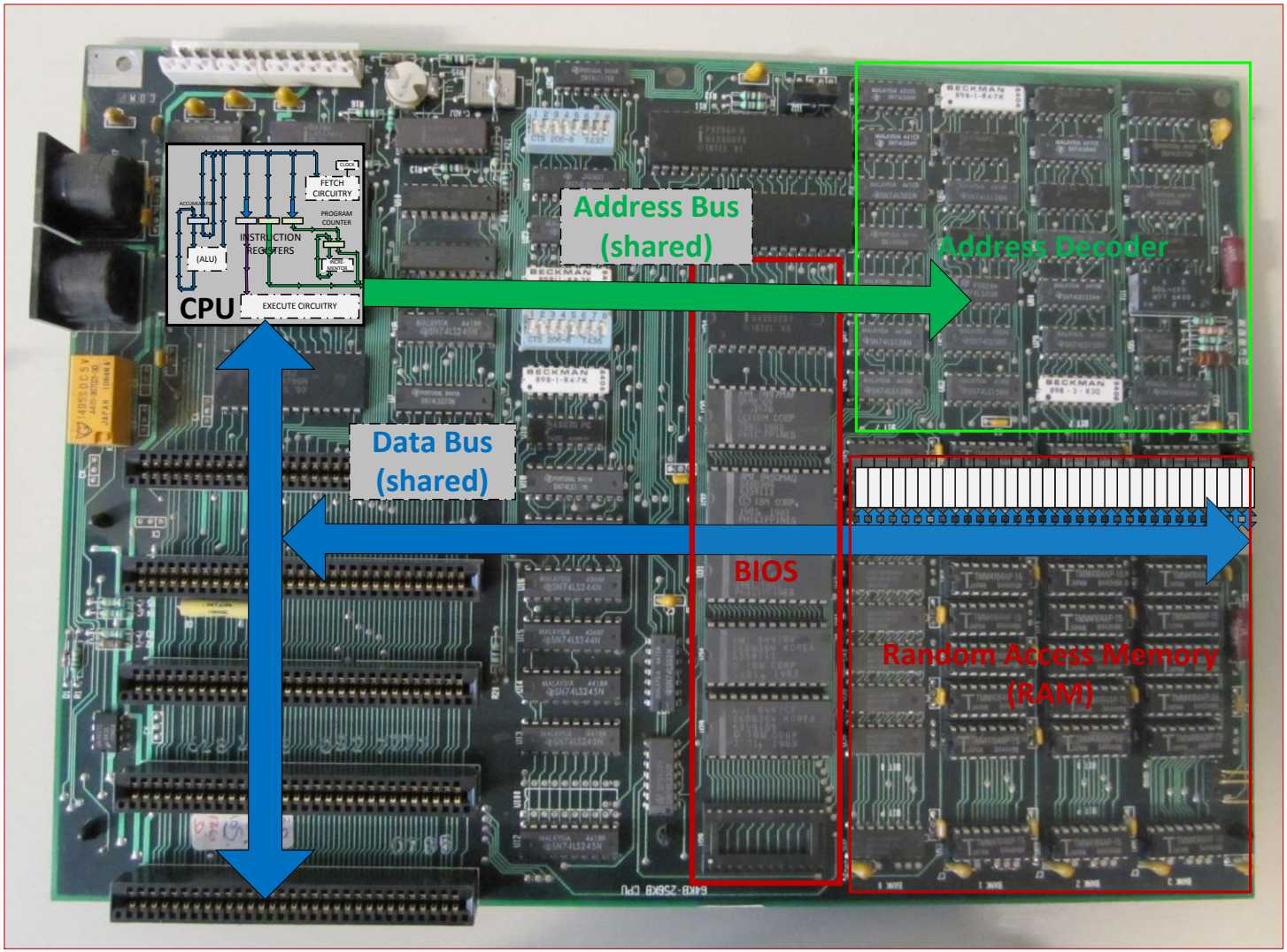
Front Side Bus Interface, 400..800 MHz

(11) ROB Reorder Buffer 3x42 entries
(12) 8 kByte Level 1 Data cache four way set associative. 1R/1W
(13) Summed Address Index decode and Way Predict
(14) Cache Line Read / Write Transferbuffers and 256 bit wide bus to and from L2 cache

April 19, 2003 www.chip-architect.com

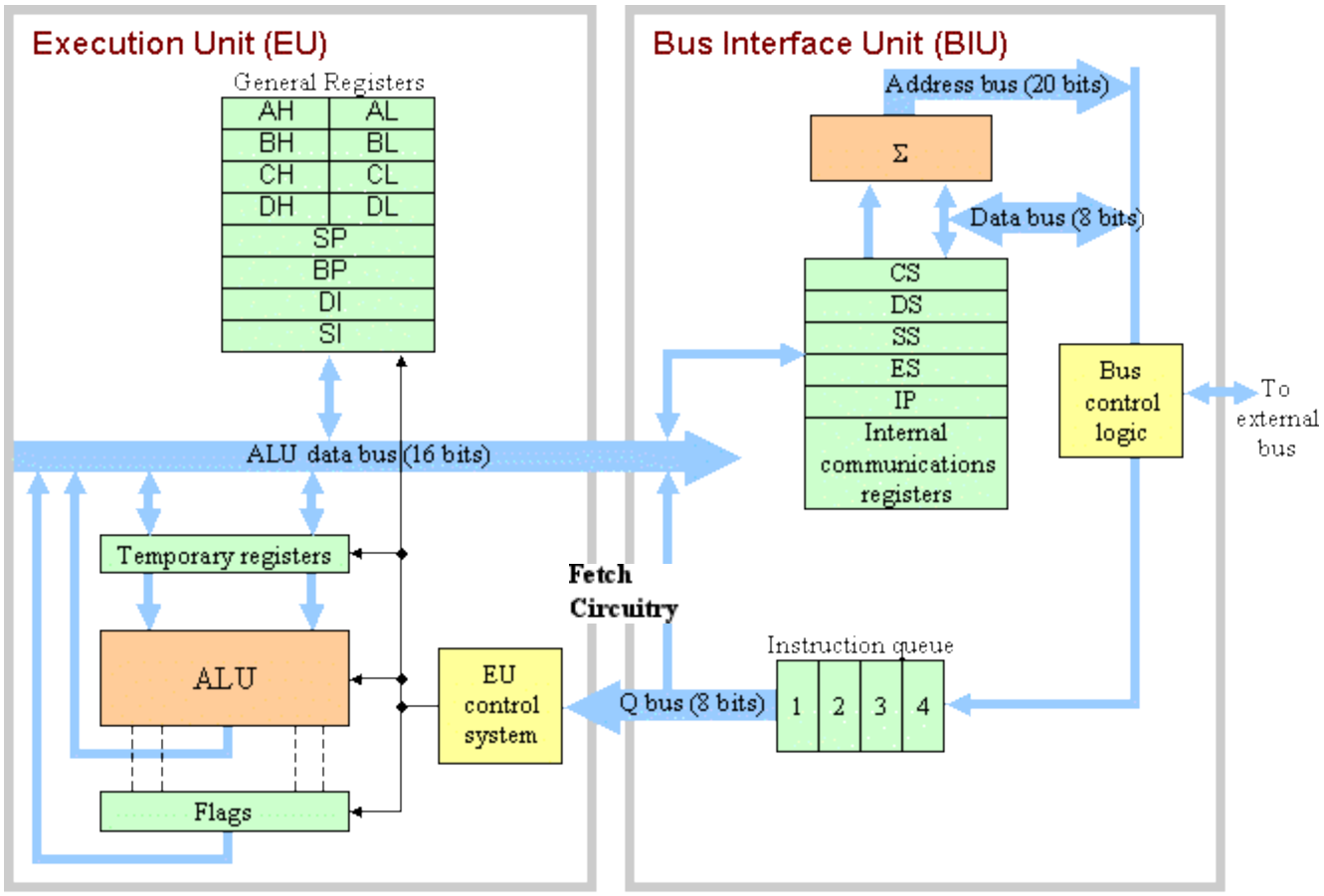


Original IBM PC Motherboard



Original IBM PC Motherboard

simplified data flow

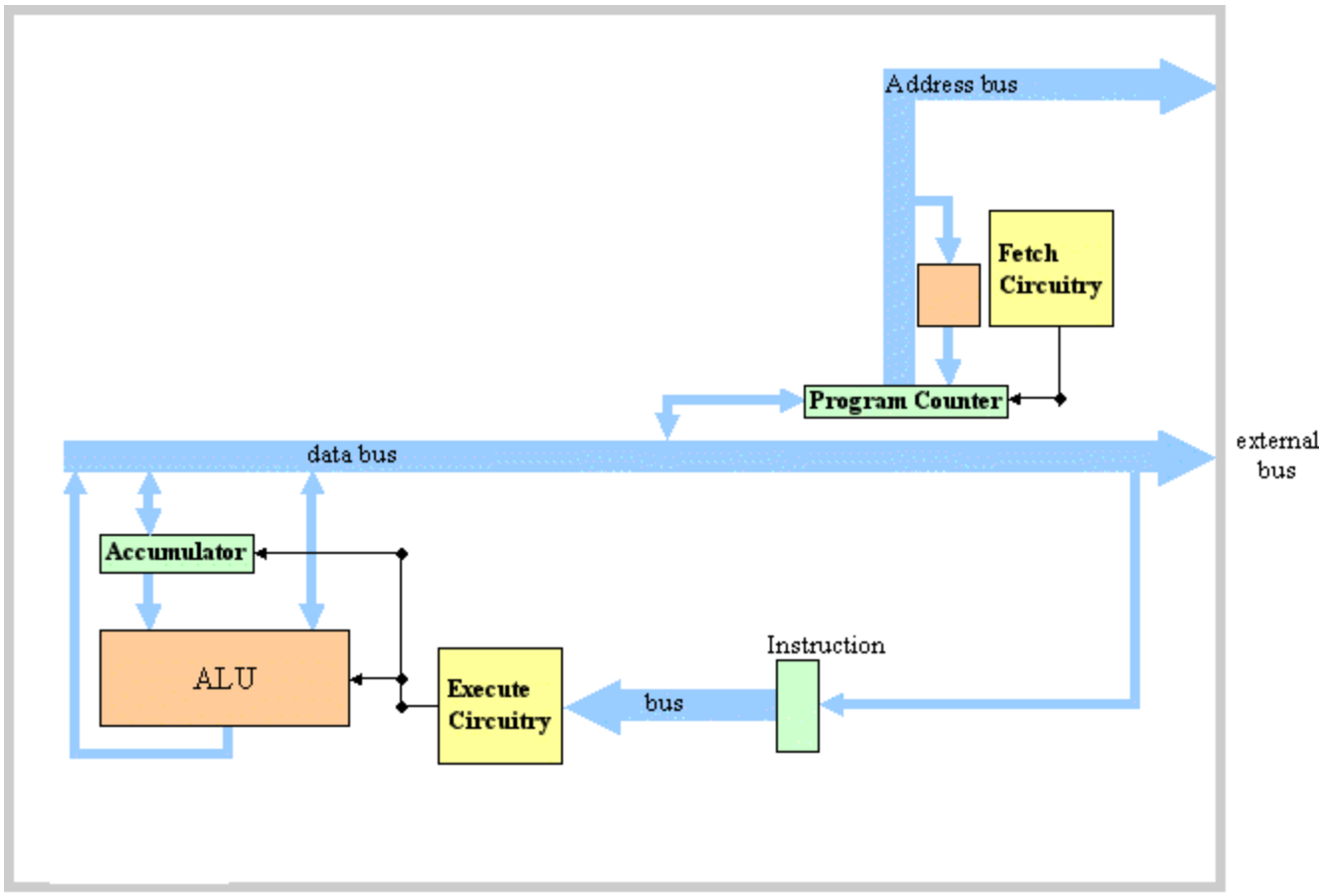


-- Static registers (groups of D Flip-Flops) used to hold or transfer binary data

-- Logic gate circuits designed to perform arithmetic or logical functions

-- Logic gate circuits designed to provide internal control to processor

-- Internal data busses used to pass information between components

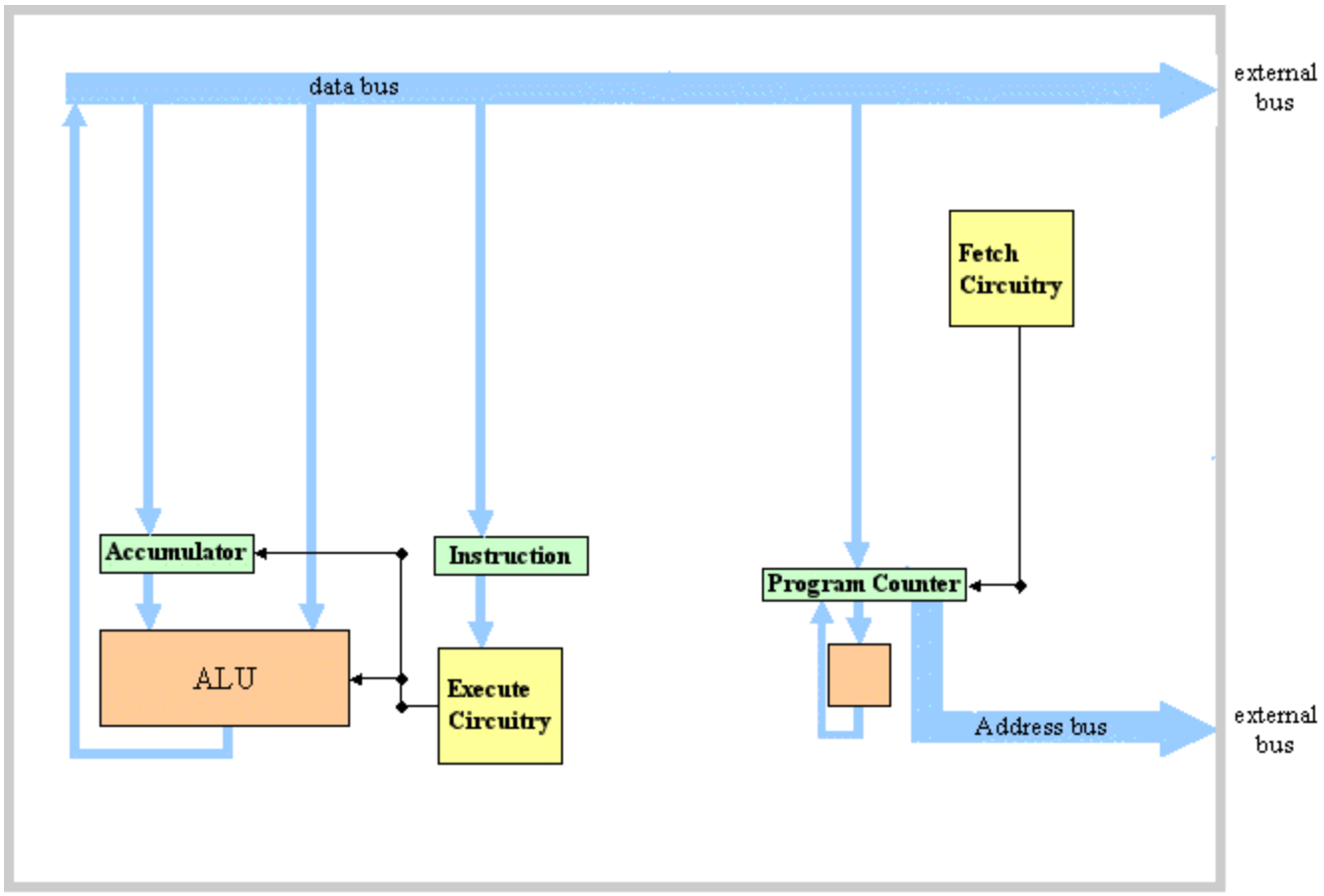


-- Static registers (groups of D Flip-Flops) used to hold or transfer binary data

-- Logic gate circuits designed to perform arithmetic or logical functions

-- Logic gate circuits designed to provide internal control to processor

-- Internal data busses used to pass information between components

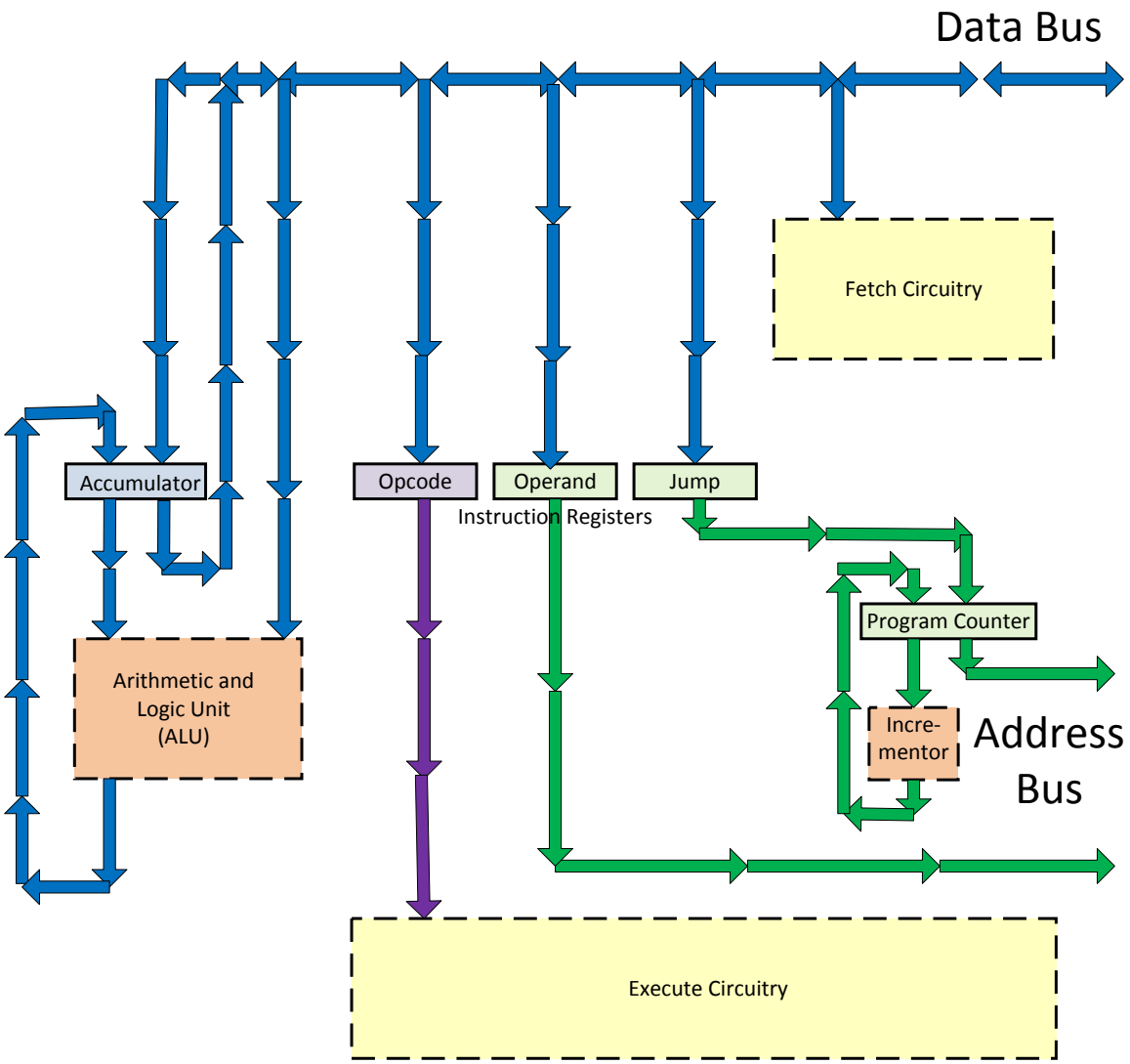


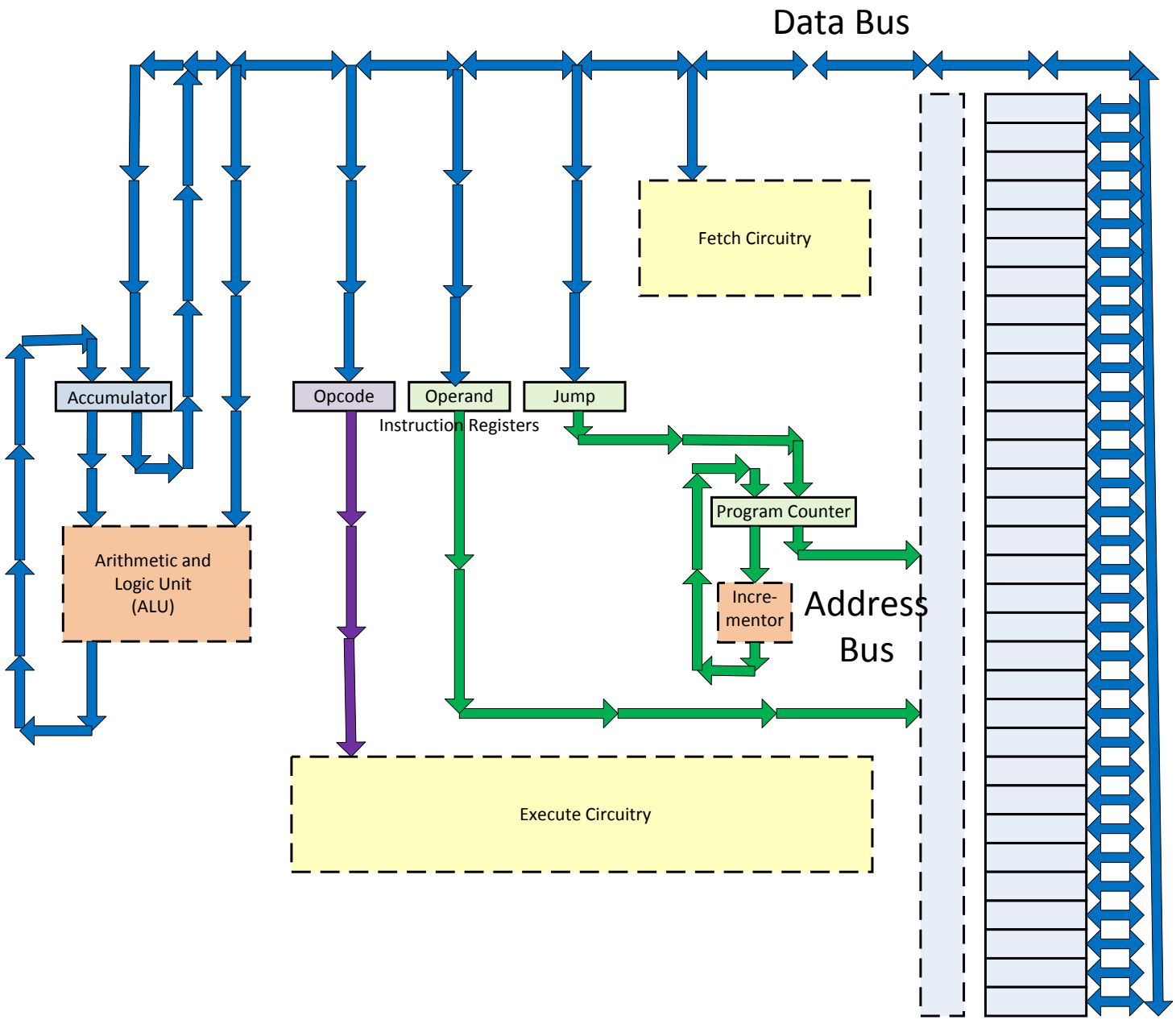
-- Static registers (groups of D Flip-Flops) used to hold or transfer binary data

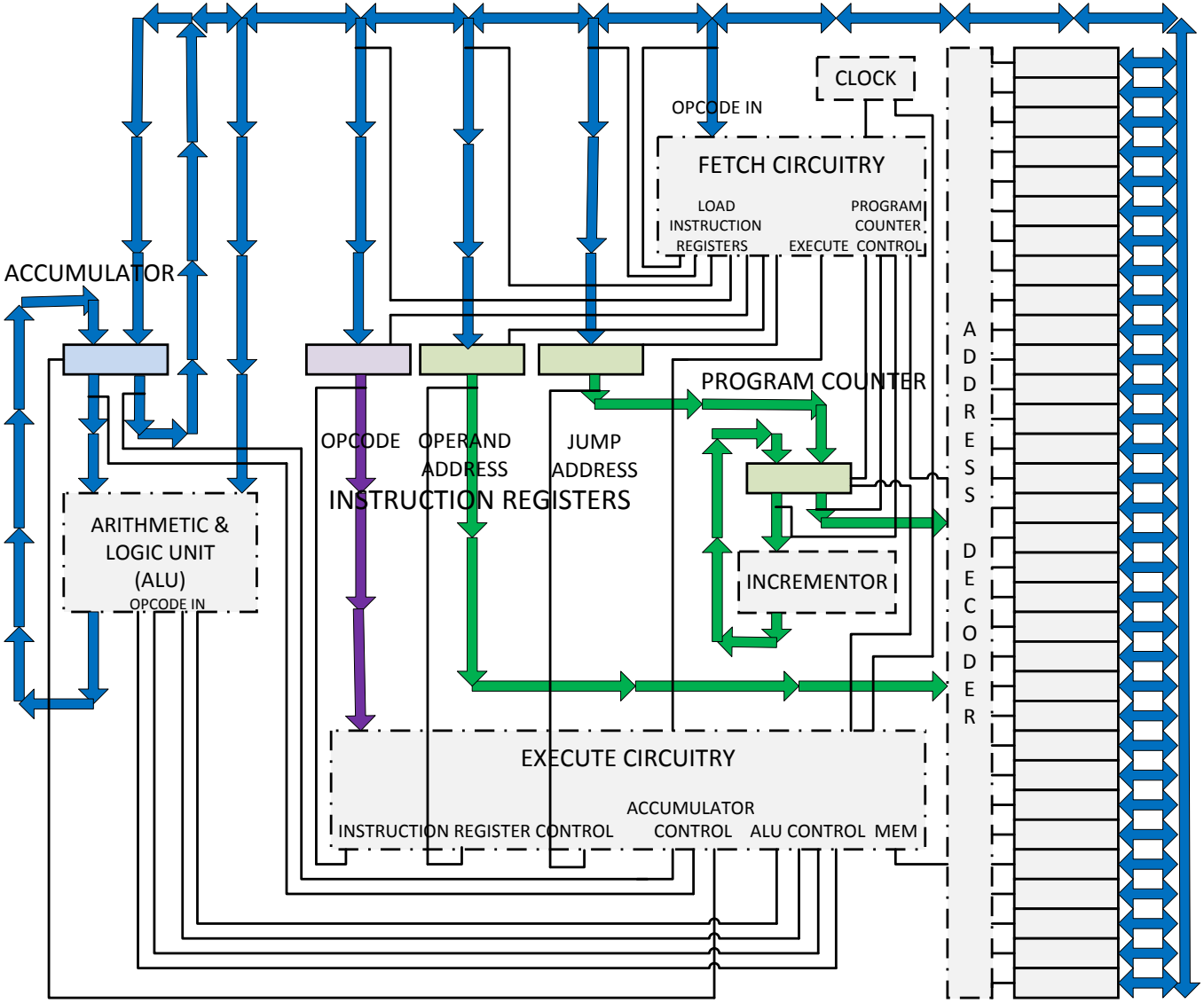
-- Logic gate circuits designed to perform arithmetic or logical functions

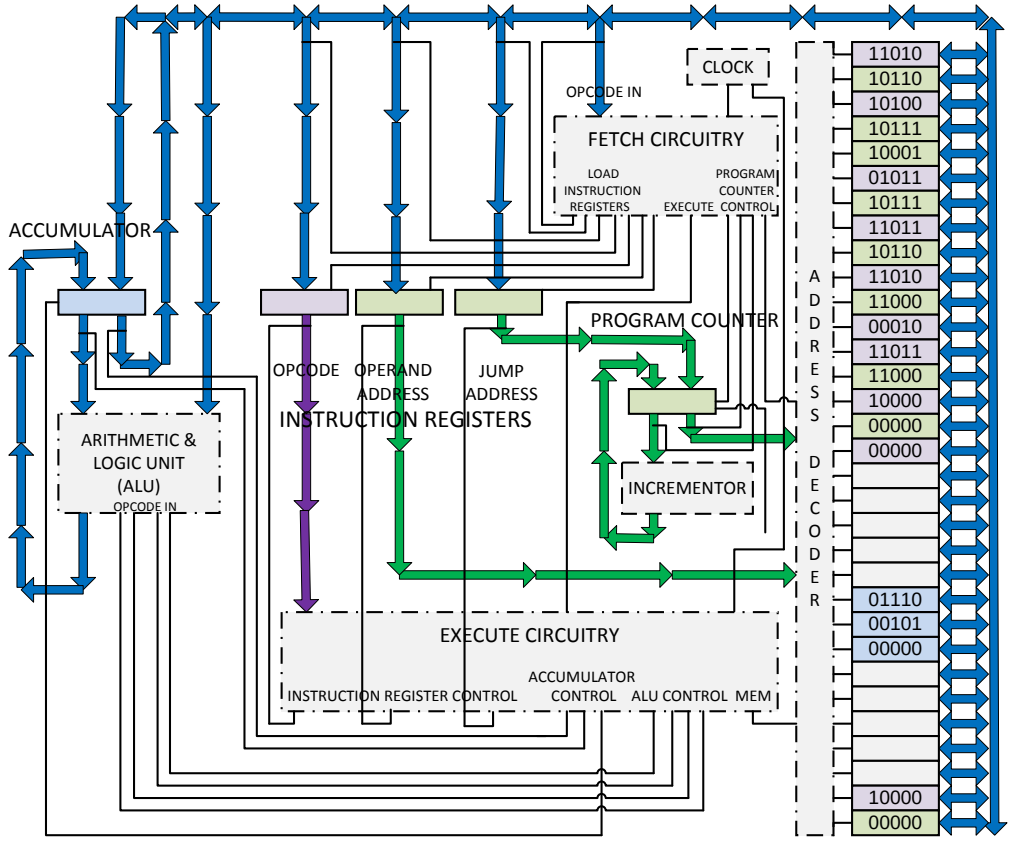
-- Logic gate circuits designed to provide internal control to processor

-- Internal data busses used to pass information between components


Data Bus
Fetch Circuitry
Accumulator
Opcode
Operand
Jump
Instruction Registers
Program Counter
Arithmetic and
Logic Unit
(ALU)
Incre-
mentor
Address
Bus
Execute Circuitry


Data Bus
Fetch Circuitry
Accumulator
Opcode
Operand
Jump
Instruction Registers
Program Counter
Arithmetic and
Logic Unit
(ALU)
Incre-
mentor
Address
Bus
Execute Circuitry


CLOCK
OPCODE IN
FETCH CIRCUITRY
LOAD INSTRUCTION REGISTERS
PROGRAM COUNTER
EXECUTE CONTROL
ACCUMULATOR
PROGRAM COUNTER
OPCODE
OPERAND ADDRESS
JUMP ADDRESS
INSTRUCTION REGISTERS
ARITHMETIC & LOGIC UNIT (ALU)
OPCODE IN
INCREMENTOR
ADDRESS DECODER
EXECUTE CIRCUITRY
INSTRUCTION REGISTER CONTROL
ACCUMULATOR CONTROL
ALU CONTROL
MEM



	Assembly Language Addresses:	Program:	Explanation	Hex
11010	Again:	LOAD	# copy the value at ‘cents’ to the Accumulator	00
10110		cents		01
10100		JLT	# if the value at ‘nickel’ is less than the Accumulator then	02
10111		nickel	# jump to the address ‘Done:’	03
10001		Done:		04
01011		SUB	# subtract the value at ‘nickel’ from the Accumulator	05
10111		nickel	# and put the result back into the Accumulator	06
11011		STOR	# copy the value in the Accumulator to ‘cents’	07
10110		cents		08
11010		LOAD	# copy the value at address ‘count’ to the Accumulator	09
11000		count		0A
00010		INC	# add 1 to the Accumulator	0B
11011		STOR	# copy the value in the Accumulator to ‘count’	0C
11000		count		0D
10000		JMP	# unconditional jump to the address ‘Again:’	0E
00000		Again:		0F
00000	Done:	HALT	# stop the processor – end of program	10
				11
				12
				13
				14
				15
01110	cents	14	# variable -- the name ‘cents’ is the address and 14 is the value	16
00101	nickel	5	# variable -- the name ‘cents’ is the address and 5 is the value	17
00000	count	0	# variable -- the name ‘count’ is the address and 0 is the value	18
				19
				1A
				1B
				1C
				1D
10000		JMP	# unconditional jump to the address ‘Again:’	1E
00000		Again:		1F

			Byte 1	Byte 2	Byte 3	
Dec	Hex	Binary	Opcode	Operand Address	Target Address	Explanation
0	00	00000	HALT			Stop CPU, no fetches
1	01					
2	02	00010	INC			Add 1 to Accumulator
3	03					
4	04	00100	NOT			Invert bits in Accumulator
5	05	00101	AND	variable		AND bits with Accumulator
6	06	00110	OR	variable		OR bits with Accumulator
7	07	00111	XOR	variable		XOR bits with Accumulator
8	08					
9	09					
10	0A	01010	ADD	variable		Add value to Accumulator
11	0B	01011	SUB	variable		Subract value from Accum.
12	0C					
13	0D					
14	0E					
15	0F					
16	10	10000	JMP	label		Unconditional jump to label
17	11					Conditional jump to label if:
18	12	10010	JEQ	variable	label	value equals Accumulator
19	13	10011	JNE	variable	label	value not equal to Accum.
20	14	10100	JLT	variable	label	value < Accumulator
21	15	10101	JGE	variable	label	value >= Accumulator
22	16	10110	JGT	variable	label	value > Accumulator
23	17	10111	JLE	variable	label	value >= Accumulator
24	18					
25	19					
26	1A	11010	LOAD	variable		Copy value to Accumulator
27	1B	11011	STOR	variable		Copy Accumulator to value
28	1C					
29	1D					
30	1E					
31	1F					

Next Presentatuion:
Execution Sequence with Calculator Pictures

End of Presentation