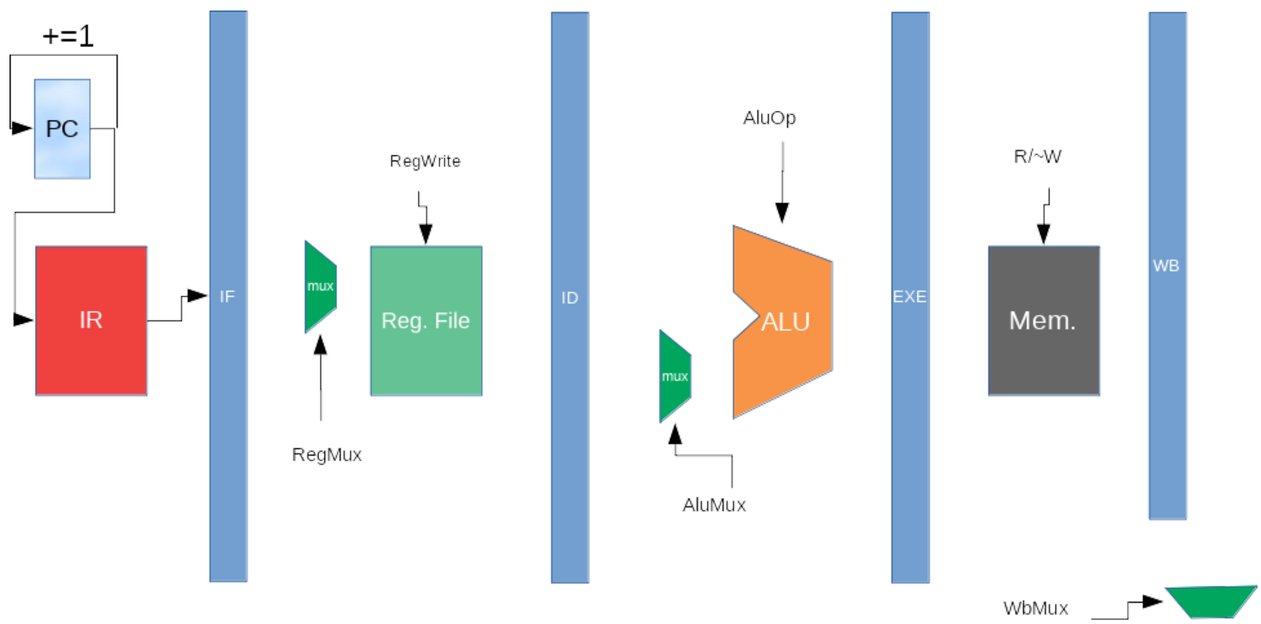
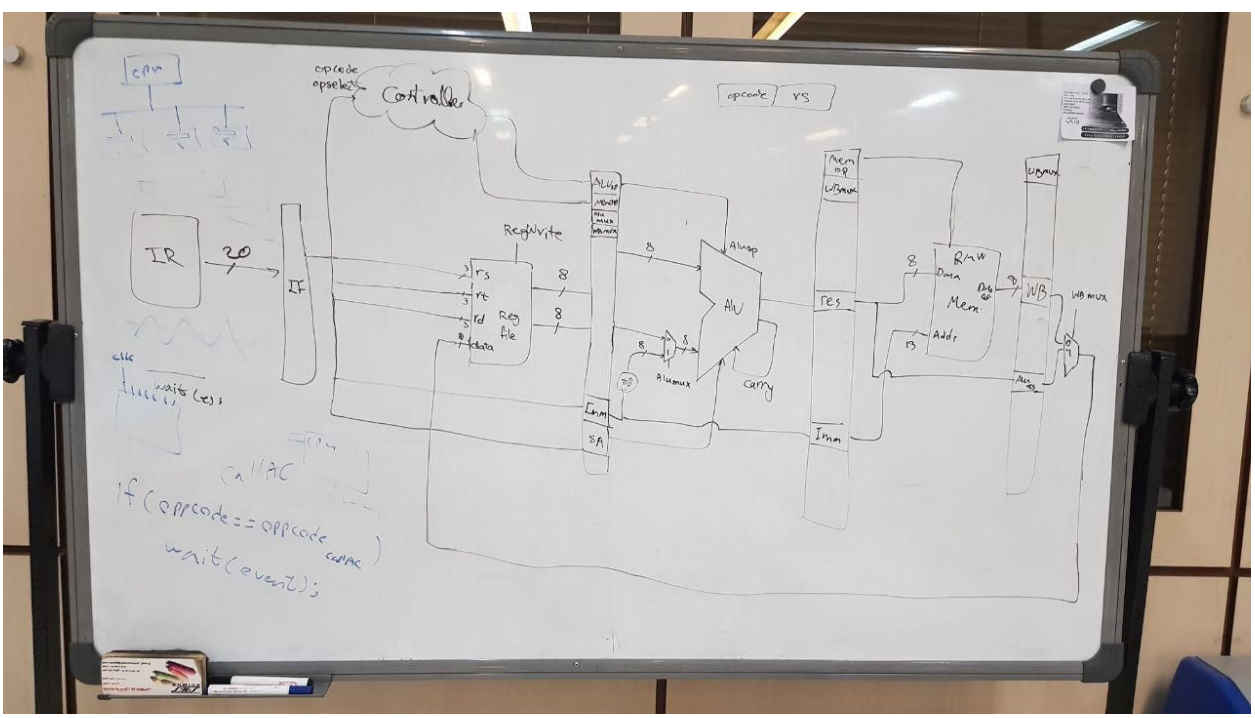
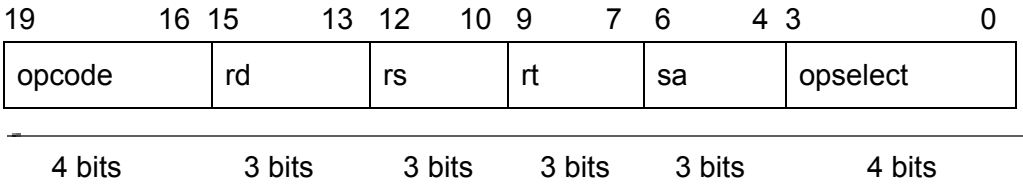
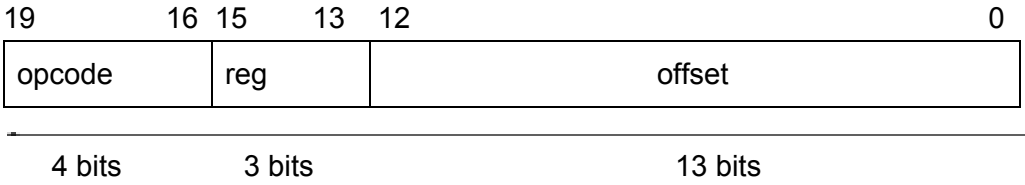

+=1
PC
IR
IF
RegWrite
mux
Reg. File
RegMux
ID
AluOp
ALU
mux
AluMux
EXE
R/~W
Mem.
WB
WbMux


Controller
IR
IF
RegWrite

R-Type



I-Type



So you can see that our Instructions need 20 bits

Operation select					
S_3	S_2	S_1	S_0	Operation	Function
0	0	0	0	F = A	Transfer A
0	0	0	0	F = A+1	Increment A
0	0	0	1	F = A+B	Addition
0	0	0	1	F = A+B+1	Add with carry
0	0	1	0	F = A+B	Subtract with borrow
0	0	1	0	F = A+B+1	Subtraction
0	0	1	1	F = A − 1	Decrement A
0	0	1	1	F = A	Transfer A
0	1	0	0	F = A ∧ B	AND
0	1	0	1	F = A ∨ B	OR
0	1	1	0	F = A ⊕ B	XOR
0	1	1	1	F = A	Complement A
1	0	×	×	F = shr A	Shift right A into F
1	1	×	×	F = shl A	Shift left A into F

ISA

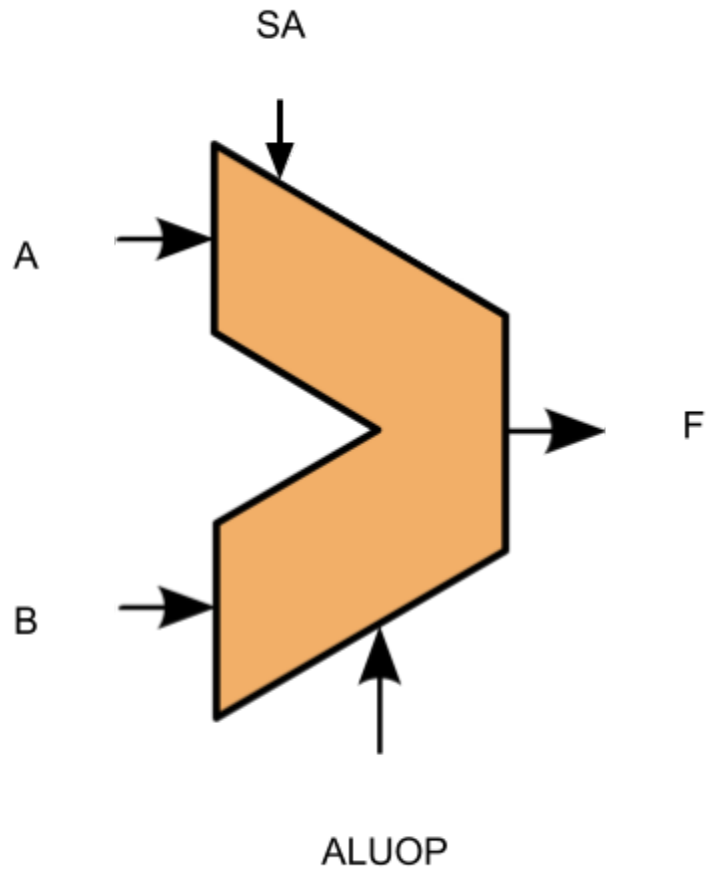
Opcode	Functionality
0000	R-Type
0001	
0010	I-type (Arithmetic) -> rs = offset [7-0]
0011	I-type (Arithmetic) -> rs += offset [7-0]
0100	LOAD
0101	STORE
0110	CALL ACC.
...	...
//	//
1111	NOP (stall)

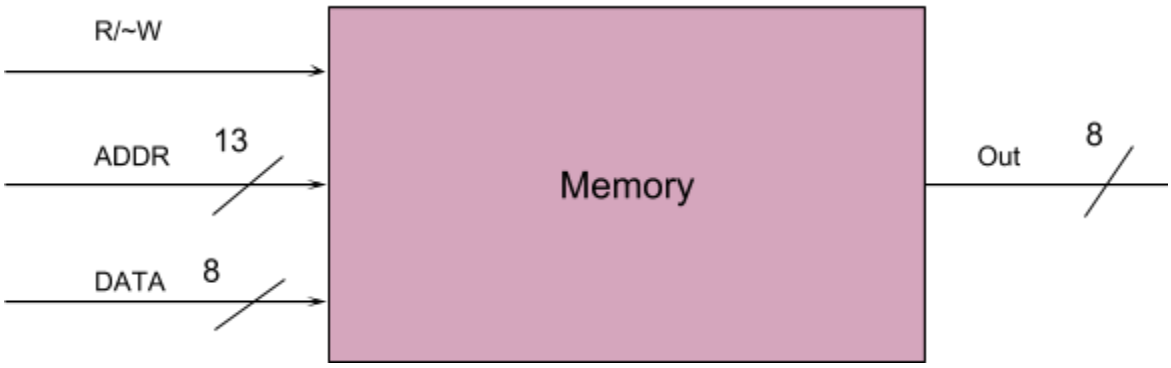
Opcode has 4 bits (but 3 bits is sufficient)

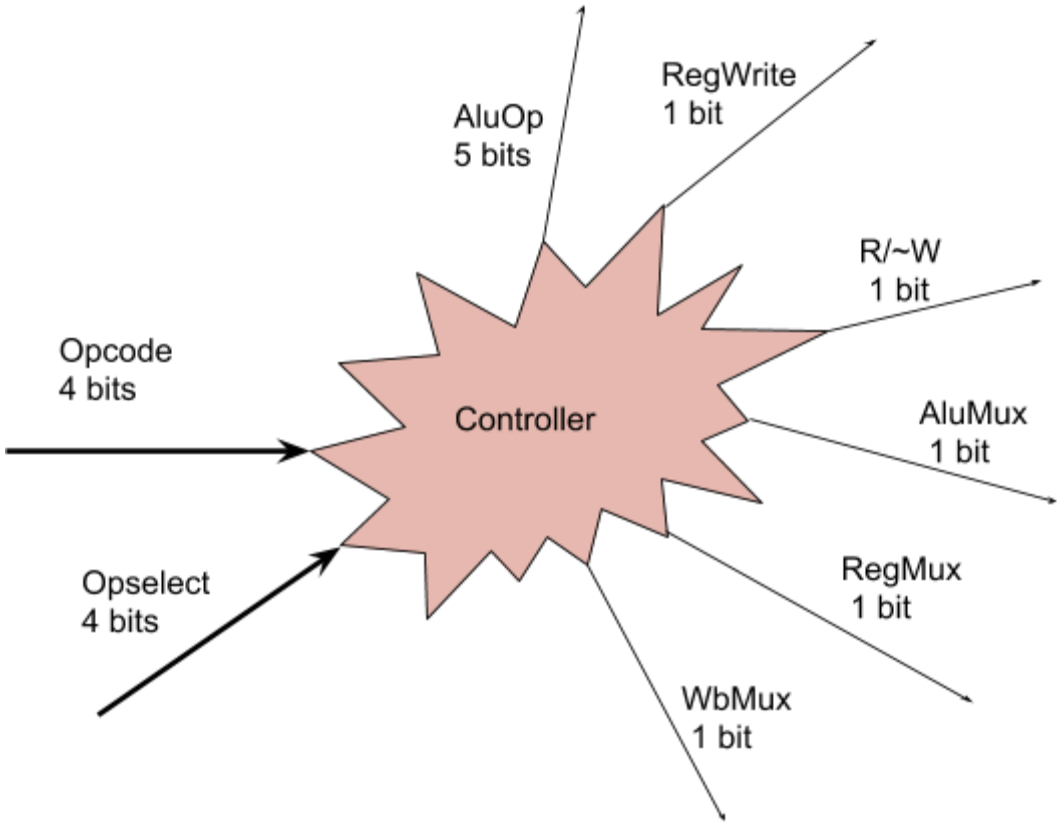
Memory (8 KB)

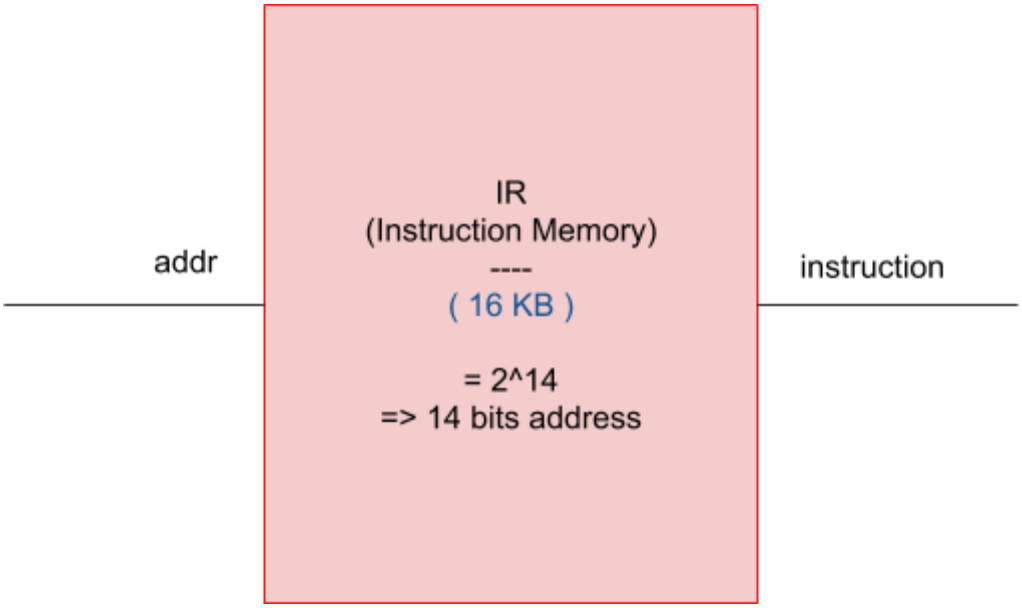
0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
1	..	..	..	..	..	..	..	..
2	..	..	..	..	..	..	..	..
...	..	..	..	..	..	..	..	..
8191 (2^13 -1)	..	..	..	..	..	..	..	..

So we need 13 bits for Addressing
And also 8 bits for data

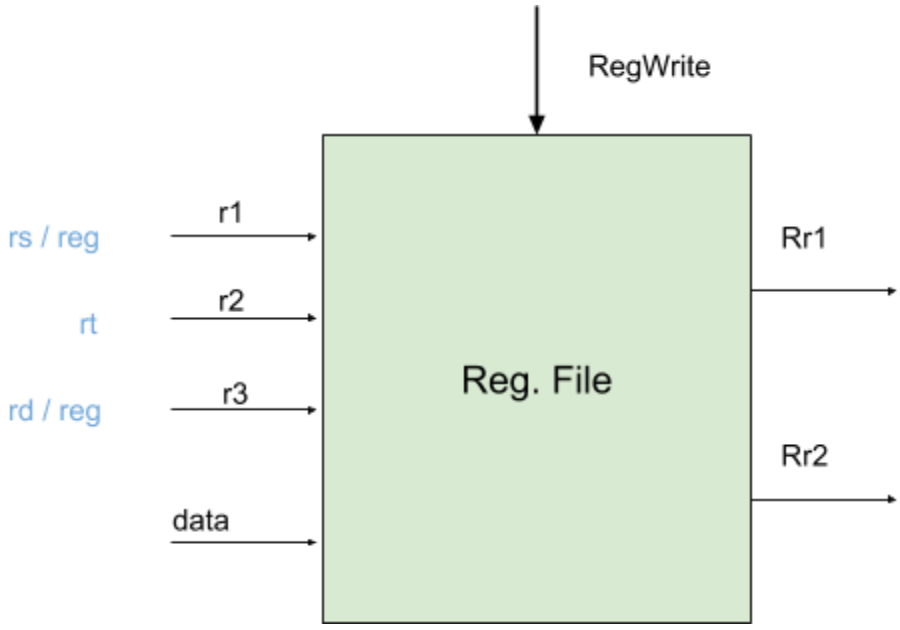

SA
A
B
F
ALUOP

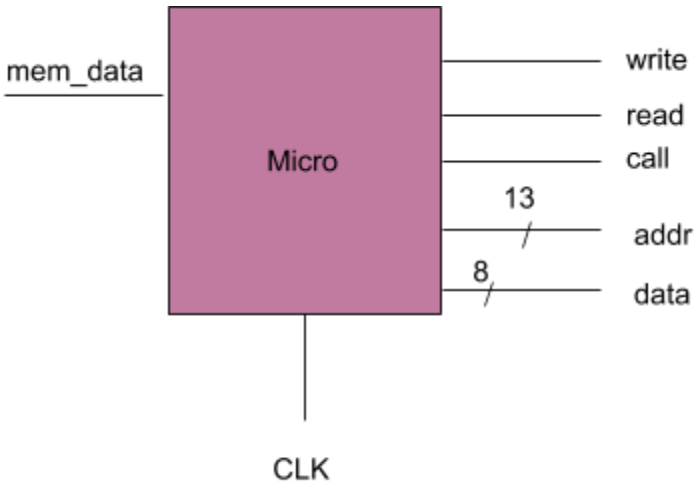

R/~W
ADDR
13
DATA
8
Memory
Out
8


AluOp
5 bits
RegWrite
1 bit
R/~W
1 bit
Opcode
4 bits
Controller
AluMux
1 bit
Opselect
4 bits
RegMux
1 bit
WbMux
1 bit


IR
(Instruction Memory)

(16 KB)
= 2^14
=> 14 bits address
addr
instruction

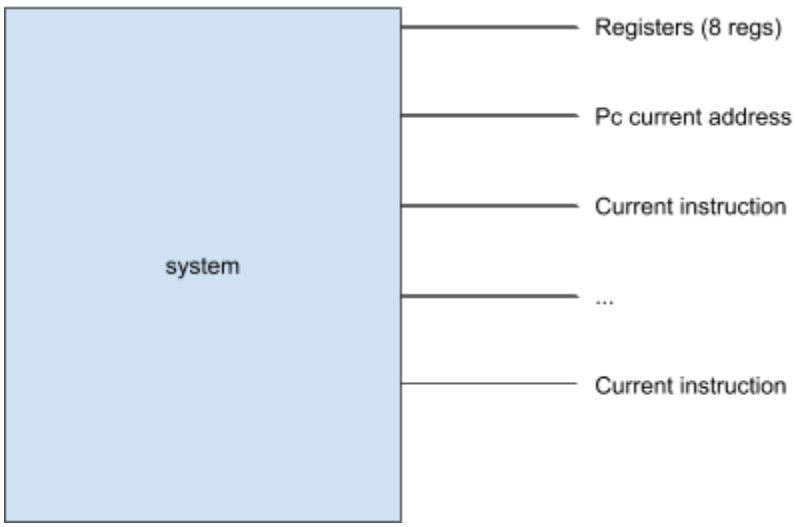

RegWrite
rs / reg
r1
r2
rt
rd / reg
r3
data
Reg. File
Rr1
Rr2


mem_data
Micro
write
read
call
13
addr
8
data
CLK

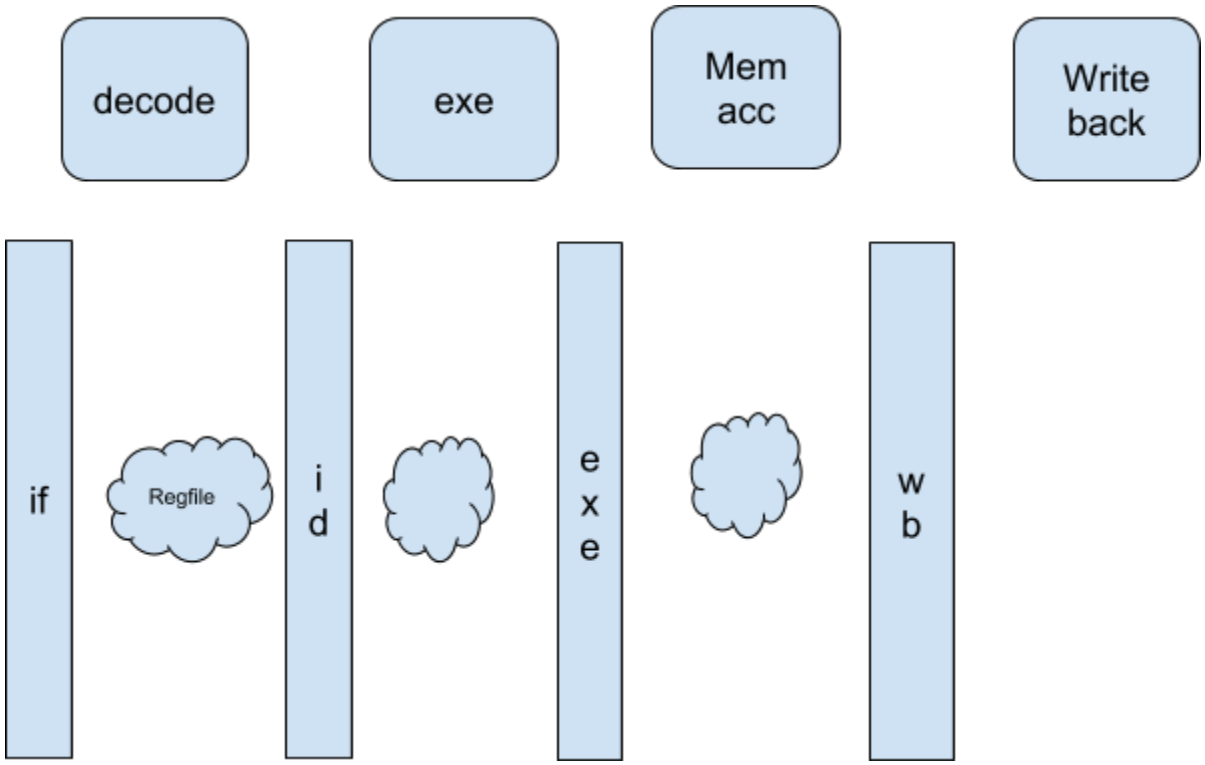
Required System C Files

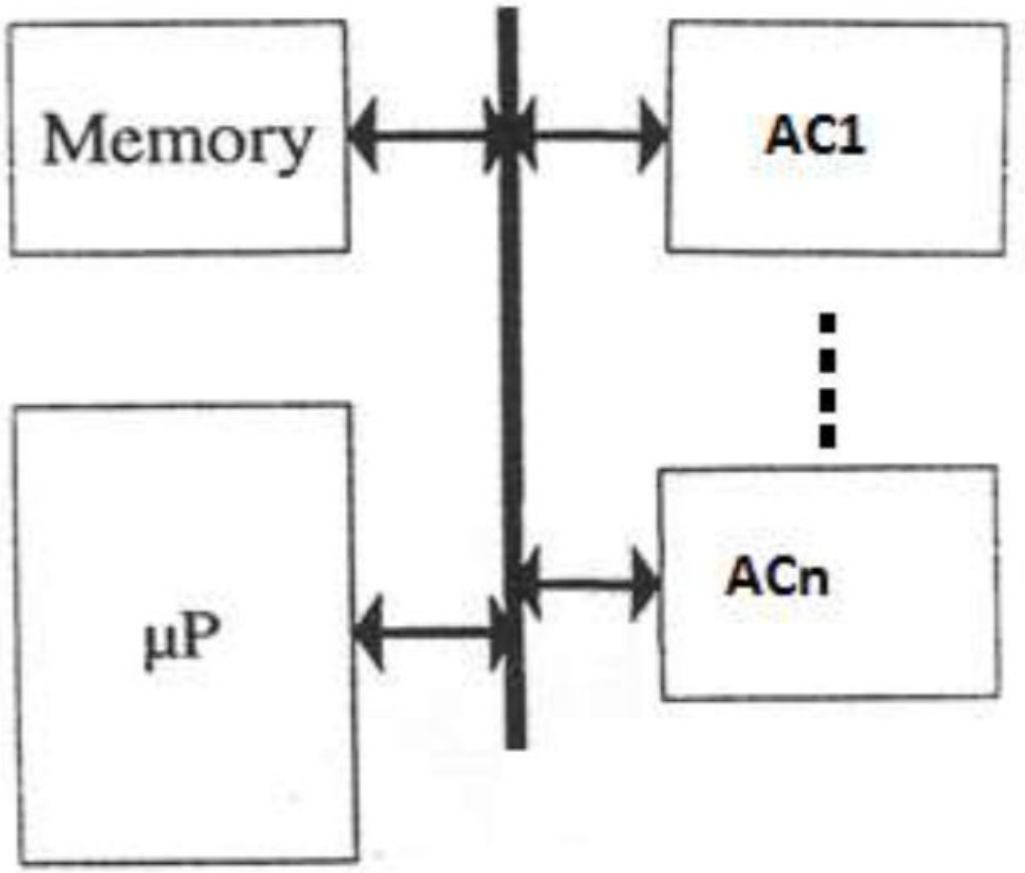
- main.cpp
- System.cpp
- Micro.cpp
- Controller.cpp
- Bus.cpp
- Register.cpp // for saving the previous data for pipelining
- Memory.cpp
- IR.cpp // instruction register -> each row has 20 bits
- RegFile // has 8 rows and each row has 8 bits
- PC.cpp // program counter
- ALU.cpp
- IF.cpp // pipeline register
- ID.cpp // pipeline register
- EXE.cpp // pipeline register
- WB.cpp // pipeline register
- Acc1.cpp ... Accn.cpp

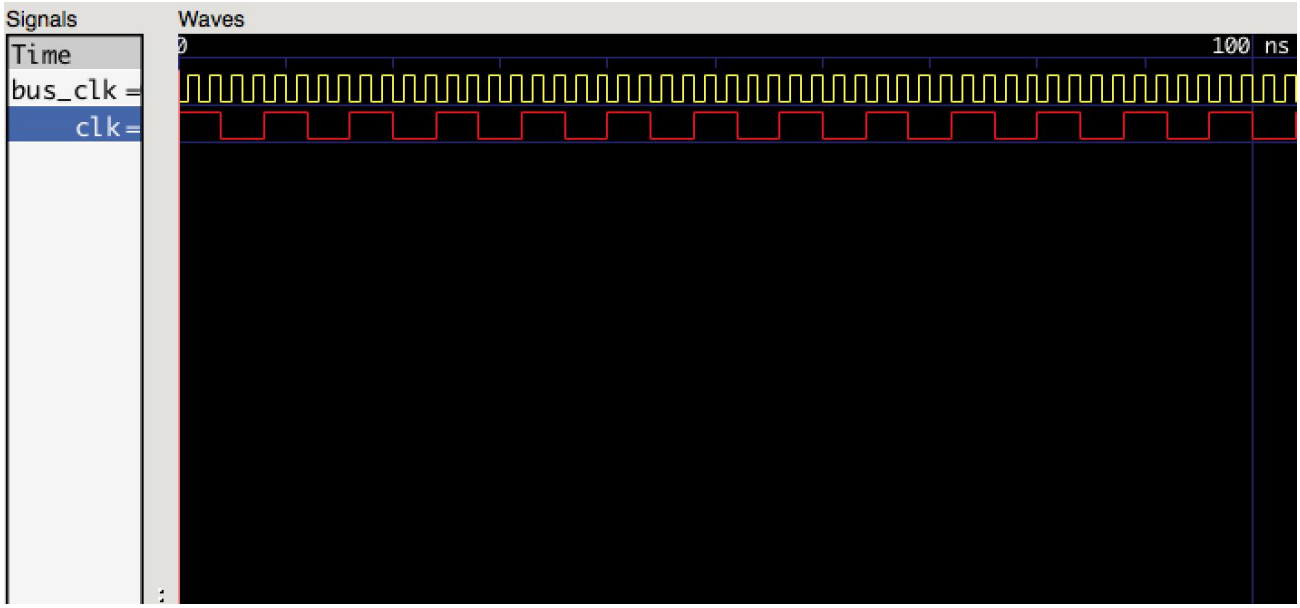
Testing wires



STALL




Memory
AC1
ACn
µP


Signals
Waves
Time
bus_clk
clk
100 ns