C88

The 8 address, 8 bit computer.
System Diagram

8 Bit Data bus
3 Bit Address bus
Other Bus
Single Wire
Operator input
Mux
Functional Unit (Sequential, Clocked)
Functional Unit (Combinational, no clock)
And gate

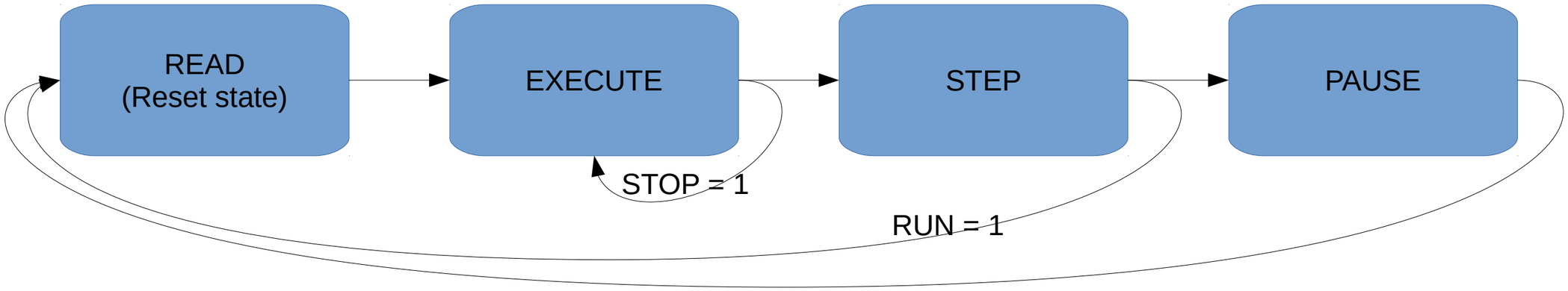
Constant "00000000"
GP Output Register
Comparator
GP Input
Main Register
A
Baddr
B
ALU
Write enable
Output register source select
Output register WEN
Data entry
PC Load source select
8x8Bit RAM
Address entry
User mode enable
PC (0-7)
Tap 2:0
Instruction Register
Decode
RAM WEN
INSTR REG WEN
Is Exec
Is Jump
STOP
Main REG WEN
Is Test
Clock Selector
Clock management
EN
Write
PC Load
PC Step
RAM Address select
Run
Step
Control FSM

C88

The 8 address, 8 bit computer.
Control FSM Specification.

Inputs:
- run (Run program, User)
- step (Single step, User)
- stop (Halt machine, Decode)
- skip (Skip next instruction, Decode & comparator)
- is_jump (This instruction is a jump, Decode)

Outputs:
- ram_addr_sel (RAM address source select)
- instr_reg_wen (Instruction register write enable)
- pc_inc (Increment program counter)
- pc_load (Load program counter)
- is_exec (Is execute state)



RUN = 1
OR STEP TRIGGERED

* The step input from user is connected to a one shot pulse generator. If the step button is held, the machine should step once.
If the button is released then pressed again then the machine will step again.

	ram_addr_sel	instr_reg_wen	pc_inc	pc_load	is_exec
READ	0	1	0	0	0
EXECUTE	1	0	skip	0	1
STEP	X	0	! is_jump	is_jump AND ! skip	0
PAUSE	X	0	0	0	0

C88

The 8 address, 8 bit computer.
Instruction set Specification.

All instructions are 8 bits wide (7 down to 0). In every instruction that requires an address, the address is in bits 2 down to 0. Bits 7 down to 3 determine the unique operation (opcode). If the operation doesn't require an address then bits 2 down to 0 are ignored and can take any value with the exception of the SHL, SHR, ROL and ROR instructions.

Mnemonic	Description	Opcode	Mnemonic	Description	Opcode	Mnemonic	Description	Opcode	Mnemonic	Description	Opcode
LOAD	Load memory location to register	00000	JMP	Jump to memory location	01000	ADD	Add memory location to register	10000	ADDU	Like ADD but unsigned	11000
SWAP	Swap a memory location with the register	00001	JMA	Jump to location specified in memory location	01001	SUB	Subtract memory location from register	10001	SUBU	Like SUB but unsigned	11001
STORE	Store the register in a memory locaion	00010	NEG	Negate the main register	01010	MUL	Multiply memory location by register	10010	MULU	Like MUL but unsigned	11010
STOP	Halt all execution	00011	INV	Invert the main register	01011	DIV	Divide register by memory location	10011	DIVU	Like DIV but unsigned	11011
TSG	Test memory location, skip if greater than register	00100	IOW	Write register to the GP ouput register	01100	SHL	Shift register left using address as constant	10100	INC	Increment register by one	11100
TSL	Test memory location, skip if less than register	00101	IOR	Copy from the GP input to the register	01101	SHR	Shift register right using address as constant	10101	DEC	Decrement register by one	11101
TSE	Test memory location, skip if equal to register	00110	IOS	Shorthand for an IOW followed by an IOR	01110	ROL	Rotate register left using address as constant	10110	DOUBLE	Double the value of register	11110
TSI	Test memory location, skip if not equal to register	00111	IOC	Clear the GP output register	01111	ROR	Rotate register right using address as constant	10111	HALF	Half the value of register	11111

C88

The 8 address, 8 bit computer.
Decoder Specification.

addr = instruction(2:0)

ALU_op = instruction(6:3)

COMP_ap = instruction(4:3)

reg_wen = 1 when (opcode >= 16)
or (opcode(4:1) = “0000”)
or (opcode = “01110”)
or (opcode = “01101”)

ram_wen = 1 when (opcode = “00001”)
or (opcode = “00010”)

stop = 1 when (opcode = “00011”)

is_jump = 1 when (opcode(4:1) = “0100”)

jump_type = opcode(0)

reg_input_select = 1 when (opcode(4:1) = “0000”)

is_test = 1 when opcode(4:2) = “001”

is_iow = 1 when (opcode = “01110”)
or (opcode = “01110”)
or (opcode = “01101”)

is_io_in = 1 when (opcode = “01110”)
Or (opcode = “01101”)

io_reg_in_sel = opcode(0)

Inputs:
- instruction (8 bit instruction from RAM)
- (opcode) (implicit, first 5 bits of instruction, 7:3)
-

Outputs:
- addr (Address specified in instruction, 3 bits)
- ALU_op (ALU operation, 4 bits)
- COMP_op (Comparator operation, 2 bits)
- reg_wen (Register write enable)
- ram_wen (RAM write enable)
- stop (Halt processor)
- is_jump (Is jump instruction)
- jump_type (Jump type, 0=direct, 1=indirect)
- reg_input_select (Register input select, 0=ALU result, 1=RAM)
- is_test (Instruction is a test and skip)
- is_iow (Instruction requires IO write)
- is_io_in (Instruction requires IO read)
- io_reg_in_sel (IO register input select, 0=Register, 1=Constant 0)

GPIO Connector spec
Female connector, outside looking in