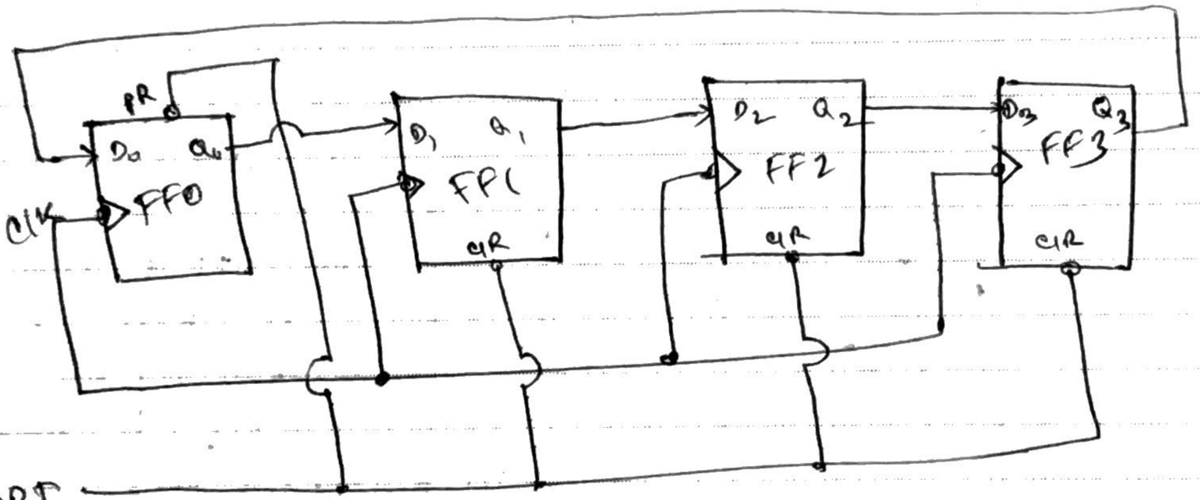
RING Counter

→ It is a typical application of shift Register

↳ The only change is the output of last ff is connected to the input of first ff.

↳ no of states = no. of flip flop



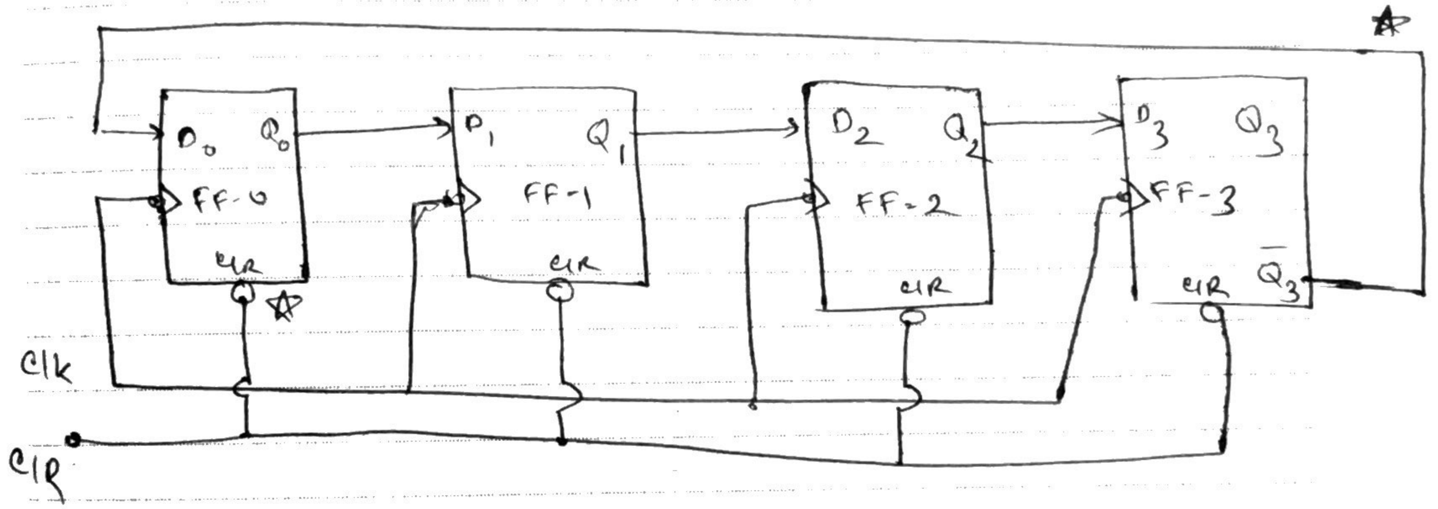
ORI
overwriting input

PRESET = 0, Q = 1
CLR = 0, Q = 0

ORI	Clk	Q0	Q1	Q2	Q3
⊔	X	1	0	0	0
1	↓	0	1	0	0
1	↓	0	0	1	0
1	↓	0	0	0	1
1	↓	1	0	0	0

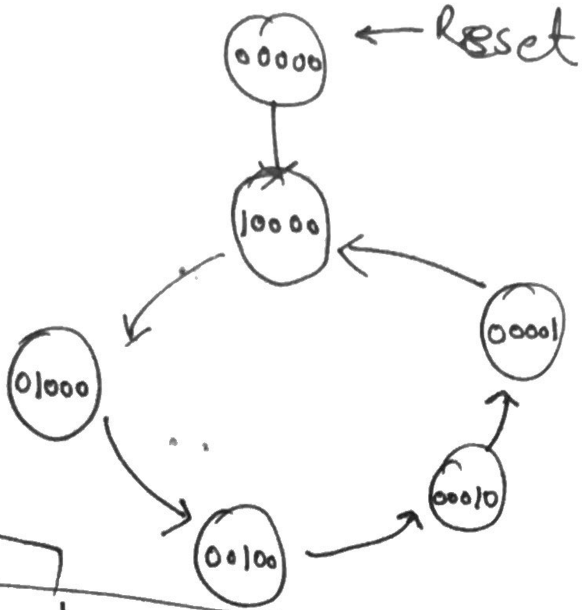
Johnson's counter (Twisted / Switch Tail Ring counter)

* no. of states = 2x number of flip flops



ClR	Clk	Q_0	Q_1	Q_2	Q_3	State no.
⊔	X	0	0	0	0	1
1	↓	1	0	0	0	2
1	↓	1	1	0	0	3
1	↓	1	1	1	0	4
1	↓	1	1	1	1	5
1	↓	0	1	1	1	6
1	↓	0	0	1	1	7
1	↓	0	0	0	1	8
1	↓	0	0	0	0	1
1	↓					
1	↓					

PRIORITY APPOINTMENTS NOTES



11000

Input

Present state					Next state					flip flop inputs				
S_1	S_2	S_3	S_4	S_5	~~S_1~~ Q_1	~~S_2~~ Q_2	~~S_3~~ Q_3	~~S_4~~ Q_4	~~S_5~~ Q_5	T_1	T_2	T_3	T_4	T_5
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0
1	0	0	0	0	0	1	0	0	0	1	1	0	0	0
0	1	0	0	0	0	0	1	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	1	0	0	0	1	1	0
0	0	0	1	0	0	0	0	0	1	0	0	0	1	1
0	0	0	0	1	1	0	0	0	0	1	0	0	0	1

$T_1 = \cancel{S_0}\,\bar{S}_2\,\bar{S}_3\,\cancel{\bar{S}}\,\bar{S}_4$

$T_2 = S_1 + S_2$

$T_3 = S_2 + S_3$

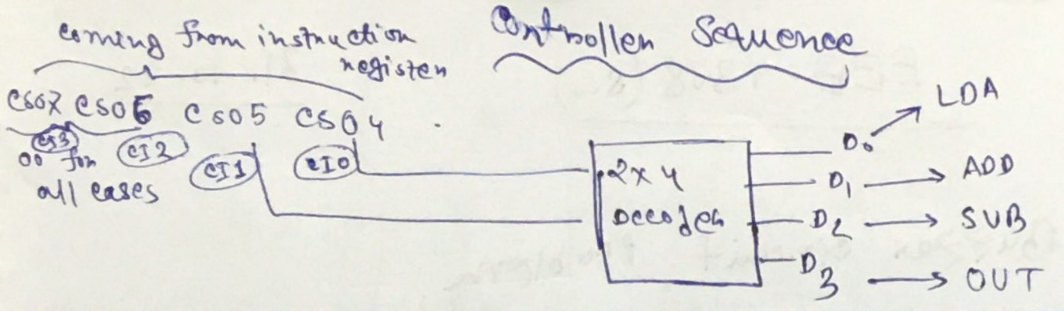
$T_4 = S_3 + S_4$

$T_5 = S_4 + S_5$

Controller Sequence

Our T states:

T_1, T_2, T_3, T_4, T_5



Decoded Instructions				States					Activated Bits											
D_3	D_2	D_1	D_0	T_5	T_4	T_3	T_2	T_1	CE	CO	MI	RO	II	IO	AI	AO	SU	EO	BI	OI
0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0
1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$CE = D_0T_2 + T_2D_1 + T_2D_2 + T_2D_3$

$CO = T_1D_0 + T_1D_1 + T_{21}D_2 + T_1D_3$

$MI = T_1D_0 + T_3D_0 + T_1D_1 + T_3D_1 + T_1D_2 + T_3D_2 + T_2D_3$

$RO = T_2D_0 + T_4D_0 + T_2D_1 + T_4D_1 + T_2D_2 + T_4D_2 + T_2D_3$

$II = T_2D_0 + T_2D_1 + T_2D_2 + T_2D_3$

$IO = T_3D_0 + T_3D_1 + T_3D_2 +$

$AI = T_4D_0 + T_5D_1 + T_5D_2 +$

$AO = T_3D_3$

$SU = T_5D_2$

$EO = T_5D_1 + T_5D_2$

$BI = T_4D_1 + T_4D_2$

$OI = T_3D_3$

HLT = CS07 · CS06

Pin	Name	Function
SU	Addition/Subtraction Select	When SU=0, addition operation, when SU=1, subtraction operation will be executed.
EO	ALU Out	Puts the ALU output onto the bus. Output of ALU should be disconnected when RO is low.

Instruction Register:

Function of instruction register is to receive and store the 8-bit instruction placed on the bus from the RAM. The contents of the instruction register are then split into two nibbles (nibble means 4-bit). The upper nibble goes into the controller-sequencer while the lower nibble should be sent to the bus. A reset pin is required for resetting the instruction register, when computer starts. Control bits of instruction register are-

Pin	Name	Function
II	Instruction Register In	Writes the current values of the bus into the instruction register.
IO	Instruction Register Out	Sends the stored data (only lower nibble) onto the bus. Output pins are disconnected when AO is low.

Controller-Sequencer:

Controller-sequencer generates necessary control signals for each block so that actions occur in a desired sequence. 12 control bits come out of the controller-sequencer block. These control bits determine how all other blocks will react to the next positive CLK edge. So, controller-sequencer must be designed in such a way so that correct control bits are already available to all the blocks before the next positive CLK edge. 12 control bits are –

CON = CE CO MI RO II IO AI AO SU EO BI OI

Functions of all 12 control bits are already described. Besides these 12 signals, controller-sequencer also generates a HLT (halt) signal, which can halt the process of computer by stopping the main clock.

Main Clock Signal:

Clock signal should be generated using 555 timer IC. The clock module needs to have two modes of operation available- (i) Auto and (ii) Manual. In manual mode, pressing a switch/push button will advance each clock cycle.

Specification of Instruction Set:

An instruction in the SAP-1 architecture consists of 1 byte. The upper nibble of this byte (bits 7-4) is called “opcode” and it defines the instruction. The lower nibble (bits 3-0) can be used to pass parameters to the instruction (for example a RAM Address).

Instruction = XXXX XXXX

Instruction Field (Opcode) Address Field

The computer must be able to perform the following five instructions-

Instruction	Mnemonic	Opcode	Description
Load Accumulator	LDA	0000	Load the value stored in a particular RAM address which is specified by the address field of the instruction, then store it in the accumulator.
Add	ADD	0001	Add the value stored in a particular RAM address to the value in the accumulator. RAM address is given by address field of instruction.
Subtract	SUB	0010	Subtract value stored in a particular RAM address from the value in the accumulator. RAM address is given by address field of instruction.
Output	OUT	0011	Load data from the accumulator and send that to the output register.
Halt	HLT	1111	Stop processing.

Instruction Cycle:

After each instruction is executed, program counter points to the next instruction stored in RAM. But each instruction consists of several smaller instructions called **microinstruction**. The time required by the CPU to execute one single instruction is called **instruction cycle**. For SAP-1, length of this instruction cycle is 6 time states (T states), where **T state** is the time required to complete one microinstruction.

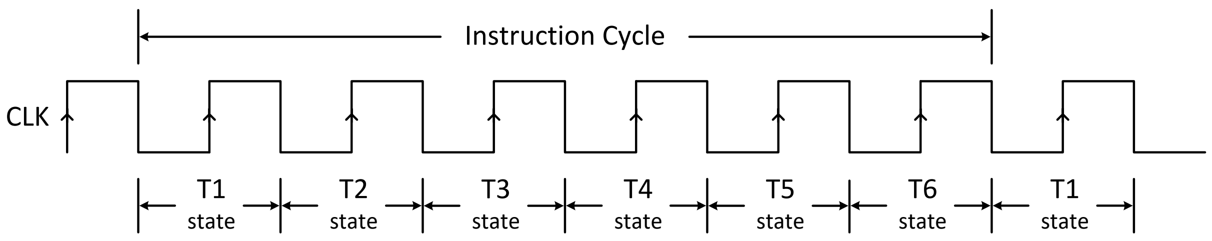


Fig. 2: T1-T6 states in SAP-1 computer.

At the beginning of a T state, controller-sequencer modifies the value of 12 control bits depending on the microinstruction. During that T state, all control bits are kept unchanged. Then, at the start of next T state, control bits are again modified based on the requirement of next microinstruction.

One important thing to observe, each T state starts at the negative edge of the clock pulse (fig. 2), where all the submodules/blocks of the computer are triggered at the positive edge of clock pulse. This is required, so that correct control bits are already available to all the submodules/blocks before the submodules trigger.

Instruction cycle can be further divided into two parts-

i. Fetch cycle (First 3 T states: T1, T2, T3)
ii. Execution cycle (Last 3 T states: T4, T5, T6)

Fetch Cycle:

Fetch cycle (T1, T2, T3) is responsible for fetching the instruction from memory. Fetch cycle is exactly same for all instructions. So, register transfers (or microinstructions) during T1-T3 states will be always same for SAP-1 for any instruction.

T State	Name	Activated Control Bits	Function
T1	Address State	CO MI	Address stored in the program counter (PC) is transferred to the memory address register (MAR).
T2	Increment State	CE	Program counter (PC) is incremented.
T3	Memory State	RO II	Instruction stored at the addressed location of RAM is transferred to instruction register.

Execution Cycle:

During execution cycle (T4, T5, T6) fetched instruction gets executed. Microinstructions during these three states can vary depending upon the instruction that was fetched. Execution cycle for each instruction (LDA, ADD, SUB, OUT, HLT) are as follows-

LDA Routine

T State	Activated Control Bits	Function
T4	IO MI	Upper nibble of instruction register output (IR) goes to controller-sequencer. Lower nibble is loaded into MAR.
T5	AI RO	Addressed data of RAM gets loaded into the accumulator.
T6		No operation.

ADD Routine

T State	Activated Control Bits	Function
T4	IO MI	Upper nibble of instruction register output (IR) goes to controller-sequencer. Lower nibble is loaded into MAR.
T5	RO BI	Addressed data of RAM gets loaded into B register.
T6	EO AI	ALU output is loaded into accumulator.

SUB Routine

T State	Activated Control Bits	Function
T4	IO MI	Upper nibble of instruction register output (IR) goes to controller-sequencer. Lower nibble is loaded into MAR.
T5	RO BI	Addressed data of RAM gets loaded into B register.
T6	EO AI SU	SU bit sets up the ALU to do subtraction. Then subtraction output of ALU gets loaded into the accumulator.

OUT Routine

T State	Activated Control Bits	Function
T4	AO OI	Content of accumulator is stored in output register.
T5		No operation.
T6		No operation.

HLT Routine

HLT does not require any control routine because no registers are involved in execution of halt instruction. After HLT instruction is fetched, controller-sequencer activates the HLT bit, which stops the main clock.