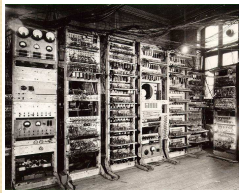
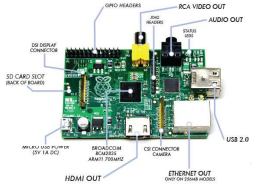
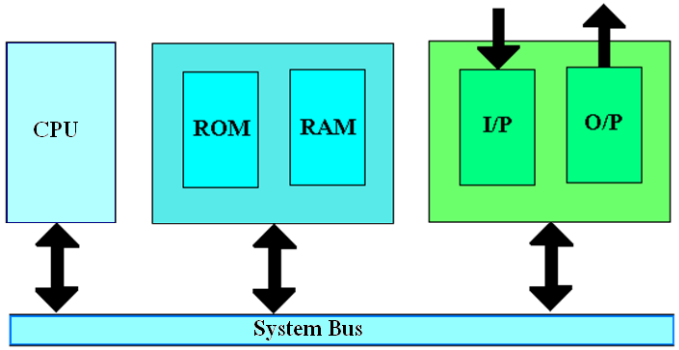
Simple Computer : SiCo





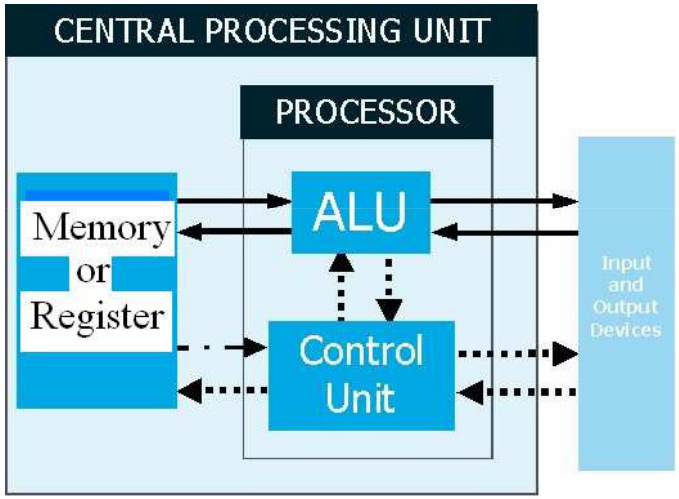
I. บทนำ ระบบคอมพิวเตอร์



CPU = Central Processing Unit
ROM = Read Only Memory
RAM = Random Access Memory

I/O = Input/Output
I/P = Input
I/O = Output

CPU : **Central processing unit**

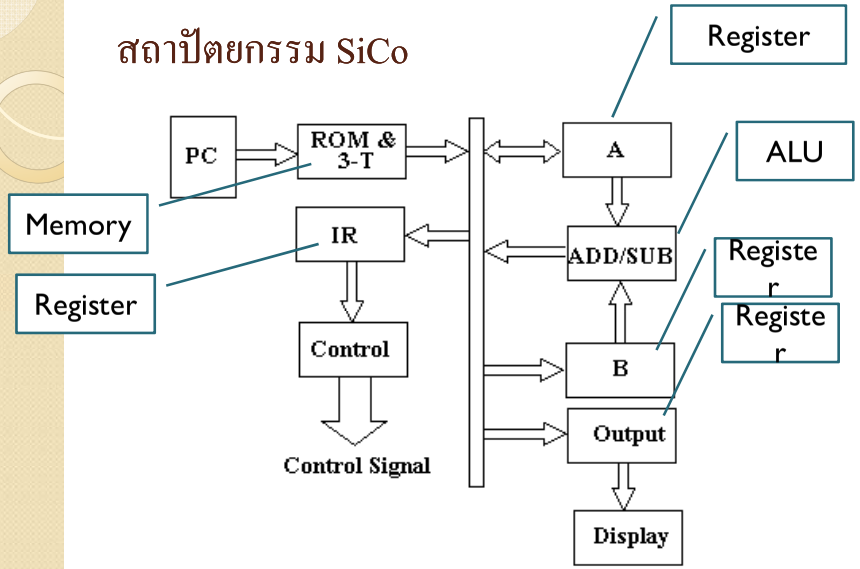


2. สถาปัตยกรรม SiCo คอมพิวเตอร์อย่างง่าย (Simple Computer)

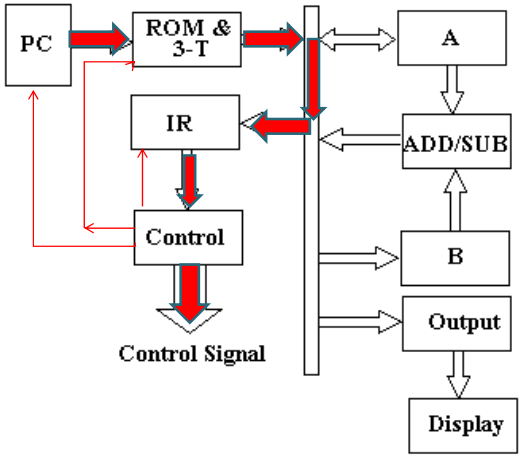
โครงสร้าง
- รีจิสเตอร์เอาท์พุท
- รีจิสเตอร์ B
- รีจิสเตอร์ A
- หน่วยประมวลผล (ALU)
- Program Counter

- Instruction Register
- หน่วยความจำ
- คำสั่ง
- วงจรถอดรหัสคำสั่ง
- Ring Counter
- สถาปัตยกรรม SiCo
- การทำงาน

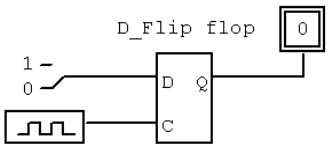
สถาปัตยกรรม SiCo

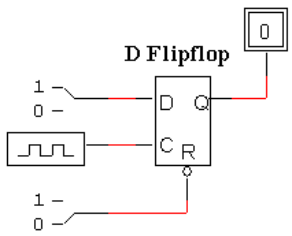


การทำงานเบื้องต้นของ SiCo



D Flipflop

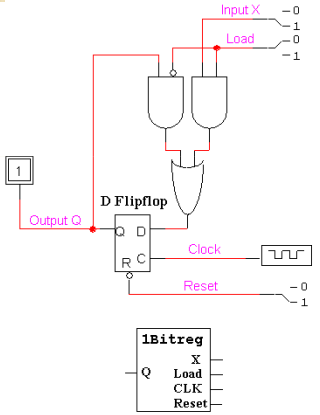




การทำงานของ ฟลิปฟลอปแบบ D

เอาท์พุท Q เปลี่ยนไปตามค่าของ อินพุท D ทุกครั้งที่มีสัญญาณ CLK (C) มากระตุ้น แต่ถ้าเป็นชนิดที่มีสัญญาณรีเซ็ท (R) เอาท์พุท Q จะเป็นโลจิก 0 ทันทีเมื่อสัญญาณที่ขา R เป็น 0 โดยไม่สนใจสัญญาณที D หรือ C แต่ถ้าสัญญาณ R เป็น 1 เอาท์พุท Q จะเปลี่ยนไปตามค่าของ อินพุท D ทุกครั้งที่มีสัญญาณ CLK (C) มากระตุ้น

1_Bit Register



เมื่อ Reset = 0 เอาท์พุท Q = 0

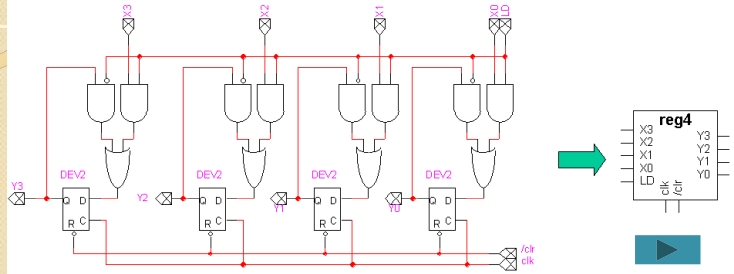
เมื่อ Reset = 1 การทำงานเป็นดังนี้

เมื่อมีสัญญาณนาฬิกา C

ถ้า Load = 0 เอาท์พุท Q คงค่าเดิม

ถ้า Load = 1 เอาท์พุท Q = Input X

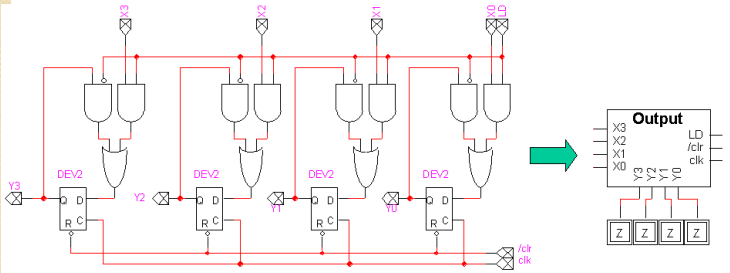
4_Bit Buffer Register



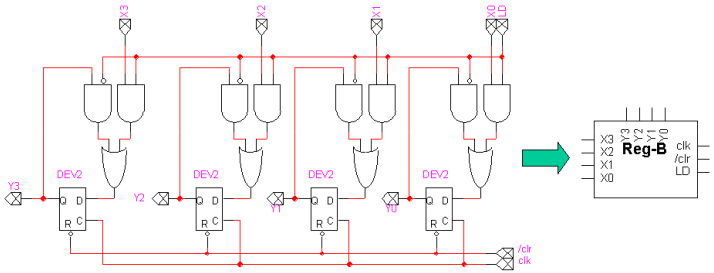
ถ้า /CLR = 0 เอาท์พุท Y3Y2Y1Y0 = “0000”
ถ้า /CLR = 1 การทำงานขึ้นอยู่กับ CLK และ LDโดยเมื่อมี CLK มากระตุ้น
ถ้า Load = 0 Y3 - Y0 จะคงค่าเดิม
ถ้า Load = 1 Y3 = X3, Y2 = X2, Y1 = X1 และ Y0 = X0

รีจิสเตอร์เอาท์พุท
(Output Register)

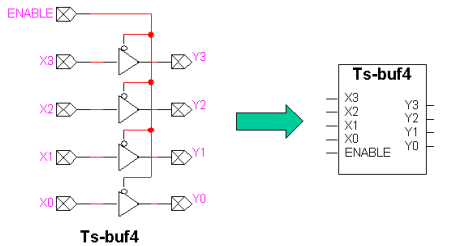
ใช้ Buffer Register ขนาด 4 บิต ทำเป็นรีจิสเตอร์เอาท์พุทสำหรับการแสดงผล



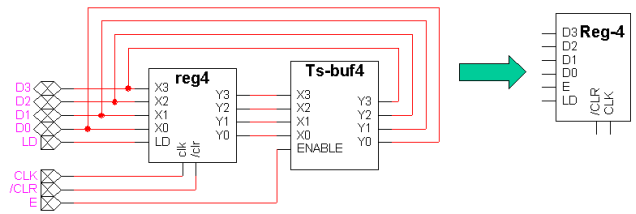
รีจิสเตอร์ B
(B Register)



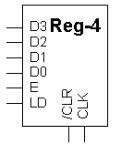
4_Bit Buffer Register with Bidirectional Bus



โดยการใช้ บัฟเฟอร์แบบ 3 สถานะ มาประกอบร่วมกับบัฟเฟอร์รีจิสเตอร์ทำให้ได้เป็นรีจิสเตอร์แบบที่มีบัส 2 ทิศทาง



การทำงานของ Bidirectional Bus Buffer Register



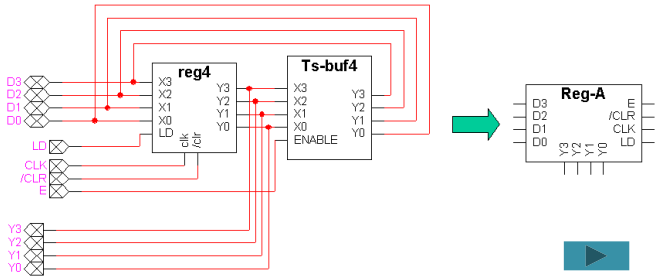
การทำงาน

ถ้า E = 1 สัญญาณเอาท์พุทของ D3, D2, D1 และ D0 จะอยู่ในสถานะ High Impedance ใช้ในสถานะรอรับข้อมูลอินพุท โดย ถ้า LD = 1 จะรับข้อมูลที่ D3-D0 ไปเก็บไว้ภายในรีจิสเตอร์ ถ้าเป็น 1 จะคงค่าข้อมูลเดิมไม่เก็บข้อมูลใหม่

ถ้า E = 0 ข้อมูลที่เก็บอยู่ภายในรีจิสเตอร์จะปรากฏออกทางสัญญาณ D3-D0

ส่วนสัญญาณ /CLR ใช้สำหรับทำให้ข้อมูลภายในรีจิสเตอร์เป็น 0 ทั้งหมด

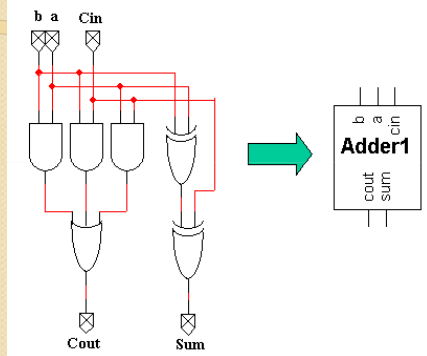
รีจิสเตอร์ A



D0-3 เป็นบัส 2 ทิศทาง

Y0 – 3 เป็นบัสทิศทางเดียว

วงจรบวก Full Adder



A

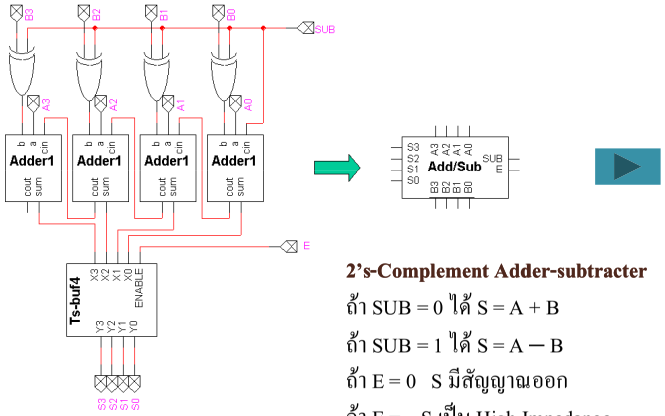
B +

C <-- Carry Input

Carry SUM

Carry Output

หน่วยประมวลผล (ALU) Arithmetic-Logic Units



2's-Complement Adder-subtracter

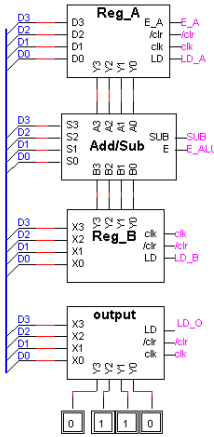
ถ้า SUB = 0 ได้ S = A + B

ถ้า SUB = 1 ได้ S = A – B

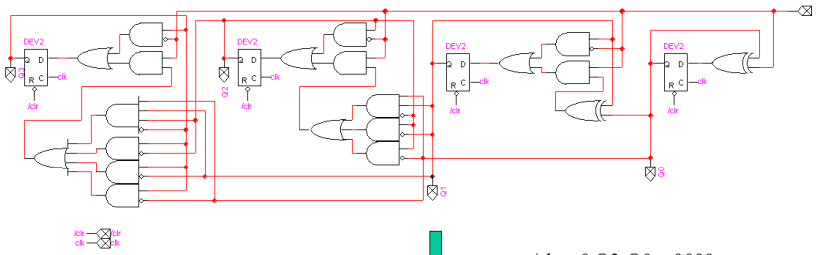
ถ้า E = 0 S มีสัญญาณออก

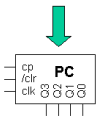
ถ้า E = S เป็น High Impedance

การต่อรีจิสเตอร์เข้ากับบัส



Program Counter (PC)





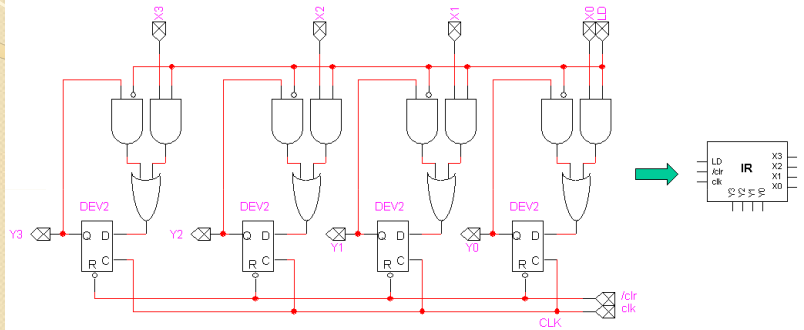
/clr = 0 Q3-Q0 = 0000

ถ้า /clr = 1 การทำงานขึ้นอยู่กับ CP และ clk

CP = 0 ค่านับคงเดิม

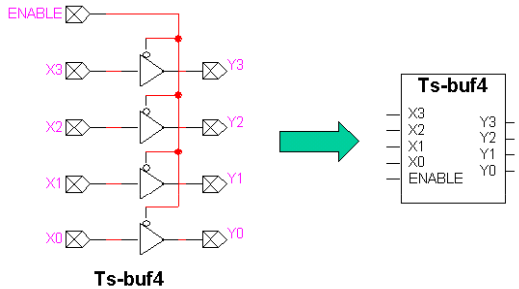
CP = 1 นับขึ้น

Instruction Register



ใช้เก็บคำสั่งระหว่างปฏิบัติงาน

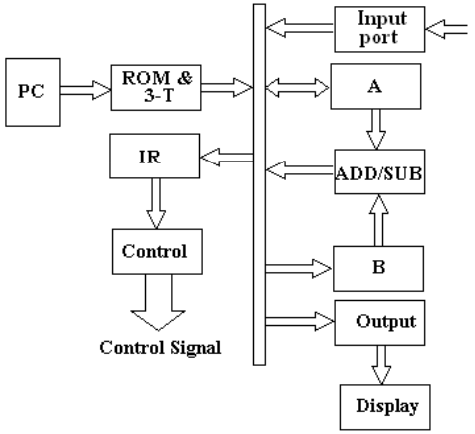
หน่วยความจำ



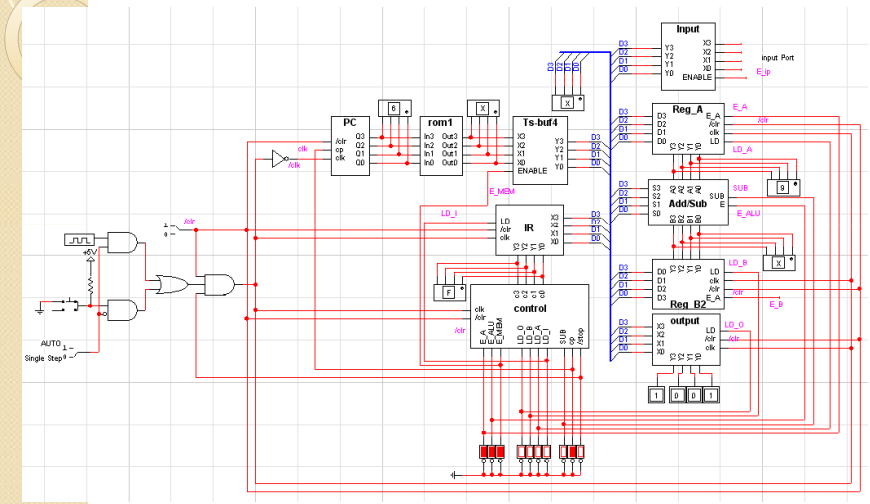
เก็บโปรแกรม

- ดู Slide 08-1-PC_ROM

SiCo 2 System



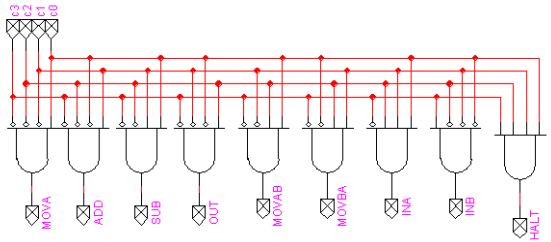
SiCo 2 System



คำสั่ง

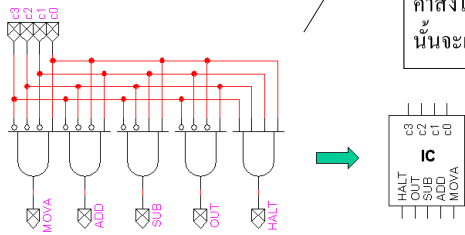
รหัส	ความหมาย	สัญญลักษณ์การทำงาน	รหัสช่วยจำ
1	เก็บข้อมูลในรีจิสเตอร์ A	A <- n	MOV A,#n
2	บวกข้อมูลในรีจิสเตอร์ A กับข้อมูล n ผลลัพธ์เก็บที่ รีจิสเตอร์ A	A <- A + n	ADD A,#n
3	ลบข้อมูลในรีจิสเตอร์ A ด้วยข้อมูล n ผลลัพธ์เก็บที่ รีจิสเตอร์ A	A <- A - n	SUB A,#n
4	ส่งข้อมูลออกแสดงผล	O/P <- A	OUT
5	เก็บข้อมูลจากรีจิสเตอร์ A ไป B	B <- A	MOV B,A
6	เก็บข้อมูลจากรีจิสเตอร์ B ไป A	A <- B	MOV A,B
7	อ่านข้อมูลจากพอร์ทอินพุทมาเก็บที่ A	A <- Input	IN A
8	อ่านข้อมูลจากพอร์ทอินพุทมาเก็บที่ B	B <- Input	IN B
F	หยุดทำงาน		HALT

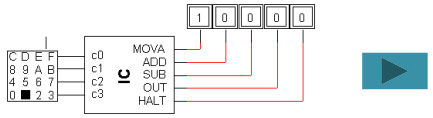
วงจรถอดรหัสคำสั่ง (Instruction Decoder)



วงจรถอดรหัสคำสั่ง (Instruction Decoder)

ถ้าสัญญาณอินพุทตรงกับคำสั่งใดสัญญาณเอาท์พุทนั้นจะเป็น โลจิก 1



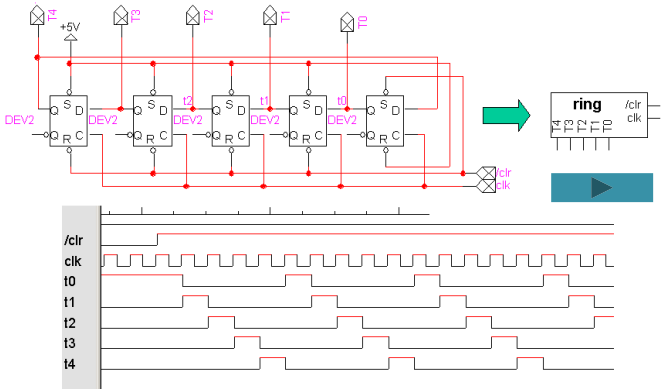


การทำงานของคำสั่ง

รหัส	รหัสช่วยจำ	การทำงาน	T0	T1	T2	T3	T4
1	MOV A,#n	A <- n	IR <- Memory	PC <- PC+1	A <- Memory	PC <- PC+1	
2	ADD A,#n	A <- A + n	IR <- Memory	PC <- PC+1	B <- Memory	PC <- PC+1	A <- (A+B)
3	SUB A,#n	A <- A - n	IR <- Memory	PC <- PC+1	B <- Memory	PC <- PC+1	A <- (A-B)
4	OUT	O/P <- A	IR <- Memory	PC <- PC+1	O/P <- A		
5	MOV B,A	B <- A					
6	MOV A,B	A <- B					
7	IN A	A <- Input					
8	IN B	B <- Input					
F	HALT		IR <- Memory	PC <- PC+1	STOP		

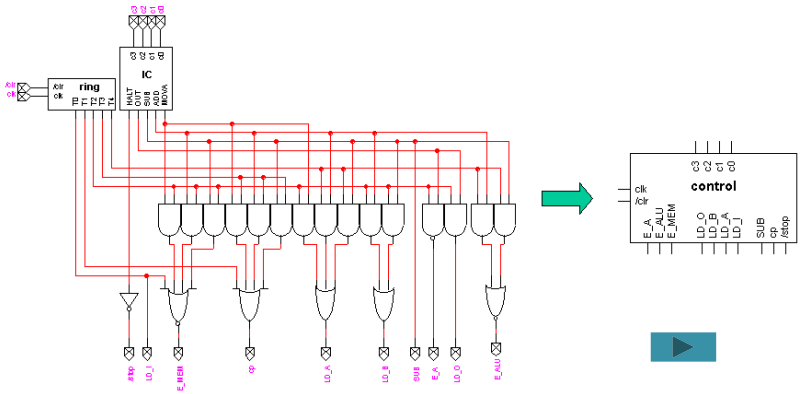
T0 ถึง T4 หมายถึงจังหวะการทำงานที่ 0 ถึง 4
ซึ่งทำได้โดยการใช้ Ring Counter

Ring Counter



ใช้สำหรับควบคุมจังหวะการทำงานของระบบ

หน่วยควบคุม (Control Unit)



ตัวอย่างโปรแกรม

		Mnemonic		
Address	Data	Opcode	Operand	Comment
		ORG	0	
0	1 5	MOV	A,#5	; Set A = 5
2	2 4	ADD	A,#4	; Add A with 4
4	4	OUT		; Display result
5	F	HALT		; Stop

หน่วยความจำ

Address	Data
0	1
1	5
2	2
3	4
4	4
5	F

ข้อมูลที่จัดเก็บในหน่วยความจำ

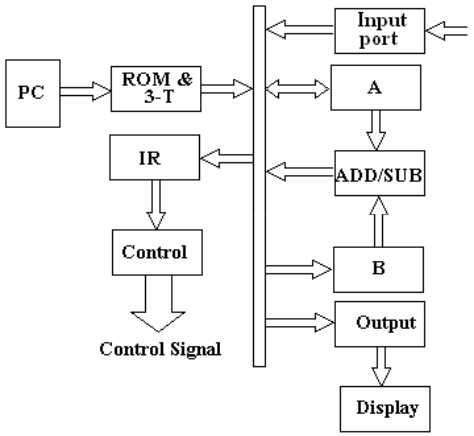
ตัวอย่างที่ 2

		Mnemonic		
Address	Data	Opcode	Operand	Comment
		ORG	0	
0	7	INA		; อ่านอินพุทมาเก็บที่ A
1	3 1	SUB	A,#1	; ลบ A ด้วย 1
3	4	OUT		; Display result
4	F	HALT		; Stop

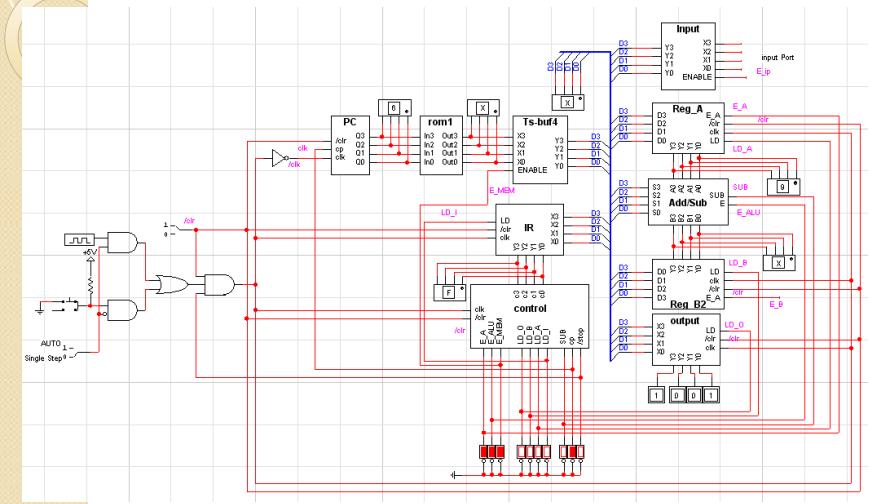
หน่วยความจำ

Address	Data
0	7
1	3
2	1
3	4
4	F

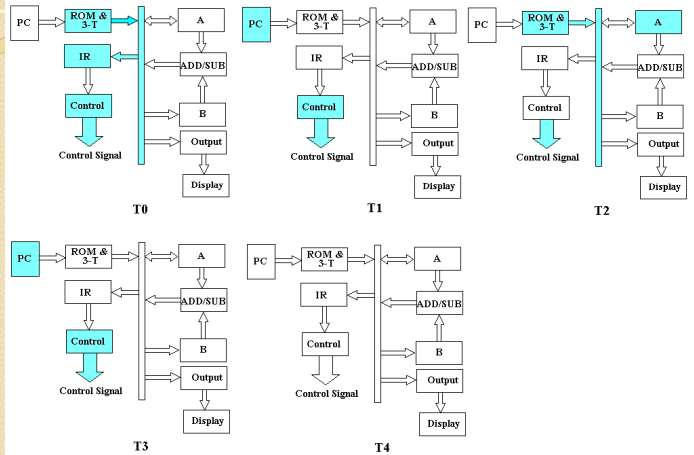
สถาปัตยกรรม SiCo



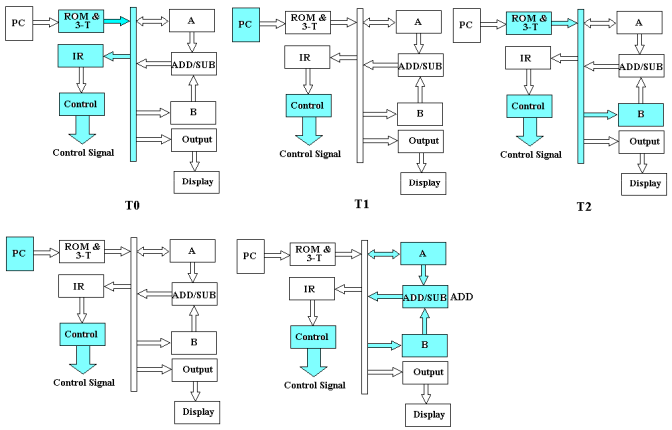
ไดอะแกรมของ SiCo



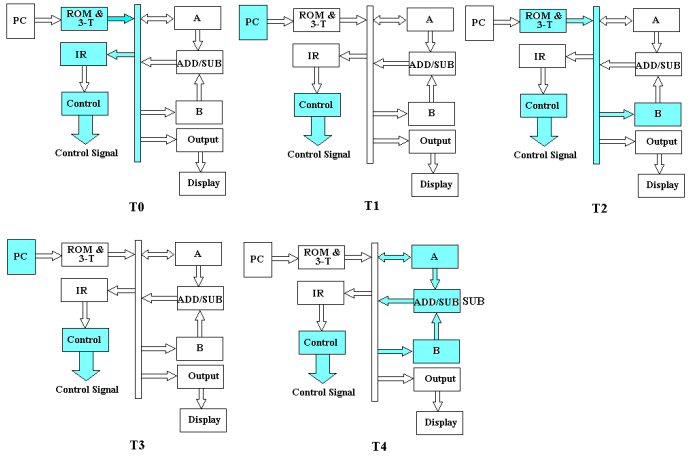
การทำงานของคำสั่ง MOV A,#n



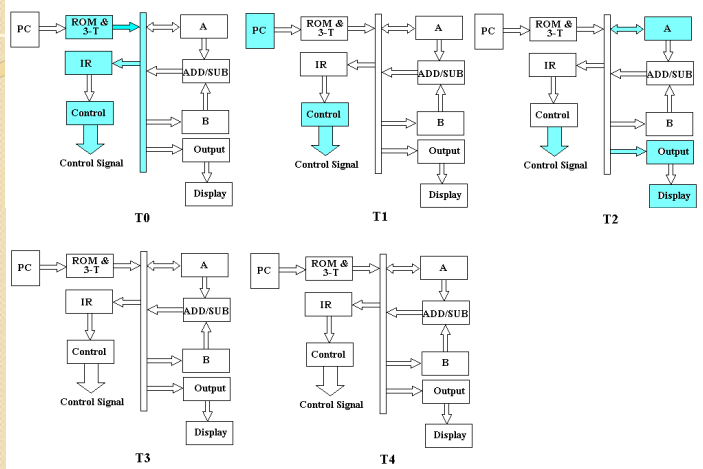
การทำงานของคำสั่ง ADD A,#n



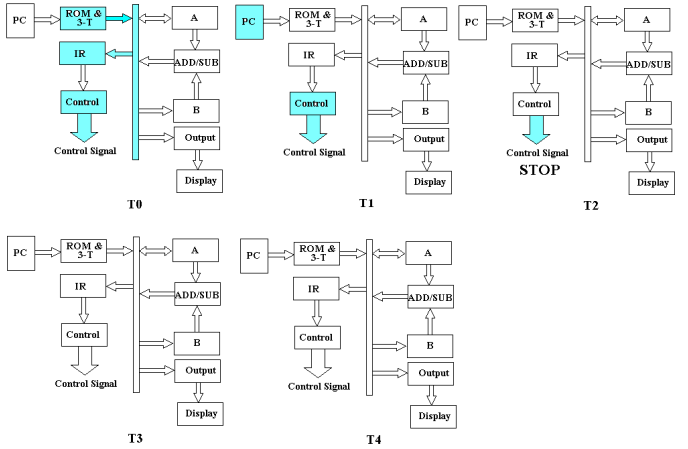
การทำงานของคำสั่ง SUB A,#n



การทำงานของคำสั่ง OUT



การทำงานของคำสั่ง HALT



ตัวอย่างโปรแกรม

		Mnemonic		
Address	Data	Opcode	Operand	Comment
		ORG	0	
0	1 5	MOV	A,#5	; Set A = 5
2	2 4	ADD	A,#4	; Add A with 4
4	4	OUT		; Display result
5	F	HALT		; Stop

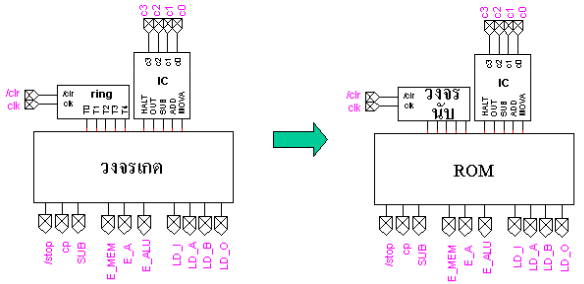
หน่วยความจำ

Address	Data
0	1
1	5
2	2
3	4
4	4
5	F

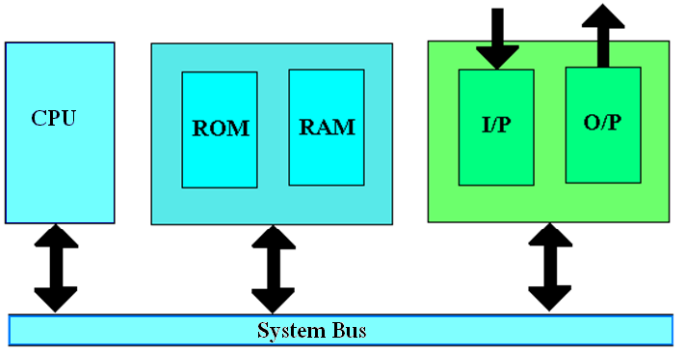
ข้อมูลที่จัดเก็บในหน่วยความจำ

Microprogramming

Microprogramming เป็นการใช้สัญญาณจาก ROM มาทำหน้าที่แทนวงจรควบคุมทำให้สามารถแก้ไขและเปลี่ยนแปลงการทำงานของวงจรได้ง่ายขึ้น



สรุปการทำงานของคอมพิวเตอร์เบื้องต้น



CPU = Central Processing Unit

ROM = Read Only Memory

RAM = Random Access Memory

I/O = Input/Output

I/P = Input

I/O = Output