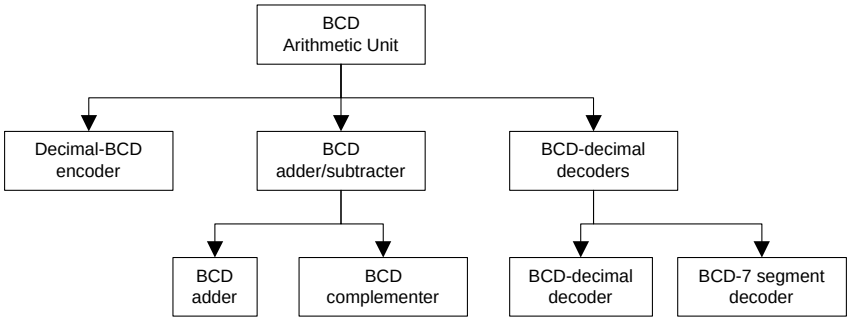
เนื้อหา

Building-Block Evolution of a Simple Computer

Hierarchical Approach

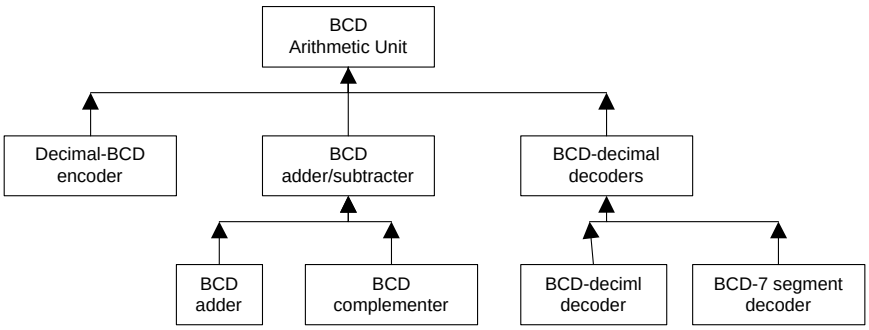
การออกแบบระบบมักจะใช้ hierarchical approach คือ ขั้นแรกจะเป็นการกำหนดความต้องการทั้งหมดของระบบ จากนั้นขั้นที่สอง จะใช้เทคนิค divide-and-conquer ในการแบ่งเป็นระบบย่อย และระบบย่อยเหล่านั้นอาจจะถูกแบ่งออกเป็น function unit ซึ่งแต่ละ unit จะมี specification ของตนเอง ดังตัวอย่างของ top-down design ของ BCD



Arithmetic unit ดังรูปที่ 1

รูปที่ 1 top-down design ของ BCD Arithmetic unit

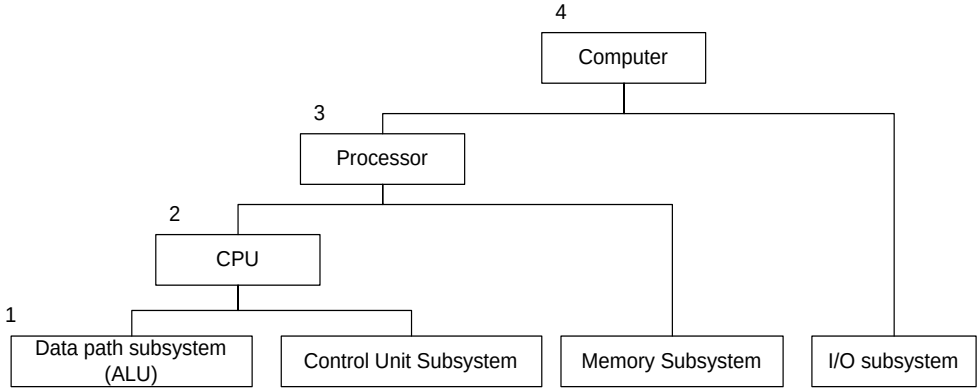
การออกแบบนี้อาจใช้วิธี bottom-up ก็ได้ คือ ถ้ามีทุกส่วนประกอบใน level ต่ำสุดแล้ว สามารถนำมารวมเป็นระบบย่อย และหลังจากมีระบบย่อยแล้ว ก็สามารถนำมารวมกันเป็นระบบทั้งหมดได้ ดังรูปที่ 2



รูปที่ 2 Bottom-Up Implementation of a System

Computer สามารถแบ่งออกได้เป็น 2 ส่วนหลักๆคือ (1.) Processor และ(2.) I/O Subsystem และ Processor ยังสามารถแยกออกเป็น 2 ส่วนคือ CPU และ Memory

Subsystem ส่วนCPUก็สามารถแบ่งออกเป็น Data Path Subsystem และ Control Unit Subsystem อีก ดังรูปที่ 3



รูปที่ 3 hierarchical organization ของ Computer แบบพื้นฐาน

รูปที่ 3 แสดงโครงสร้างของ Computer ที่มี I/O โดยวางโครงสร้างเป็น hierarchical model ที่บรรจุระดับส่วนประกอบภายในเป็น 4 ระดับ ดังนี้

1. Data Path Subsystem เป็น manual computer ที่สามารถกระทำ 2^n ฟังก์ชั่น โดยขึ้นอยู่กับการ Set ค่า Control ทั้ง n เส้น
2. Data Path Subsystem รวมกับ Control Unit Subsystem เป็น CPU (Computer กึ่งอัตโนมัติ) ที่ใช้ Opcode ของคำสั่งไปควบคม Data Processing ใน Data Path
3. CPU รวมกับ Memory Subsystem เป็น Processor ที่ทำงานภายใต้การควบคุมของ Program Instruction ที่เก็บใน Memory
4. Processor รวมกับ I/O Subsystem เป็น Computer

ดังนั้น Digital Computer แบบพื้นฐาน ก็สามารถมองเป็นระบบที่มี 4 ระบบย่อย คือ

- Data Path Subsystem
- Control Unit Subsystem
- Memory Subsystem
- I/O Subsystem

เราจะใช้ 4-step problem-solving procedure มาออกแบบ processor ของ computer

step 1 : Problem statement : ให้ instruction set ดังตารางที่ 9.1 จงออกแบบ computer ที่มีความสามารถในการโหลดและเก็บ instruction และ data ของโปรแกรม และการ execute โปรแกรมที่เก็บใน memory ได้

a. Process: Fetch (อ่านจาก memory) , decode, execute แต่ละ instruction (และทำ process ของ data ที่จำเป็น และเก็บผลลัพธ์ที่ได้)

b. Process Control Algorithm : ใช้ step-by-step procedure เพื่อทำการ fetch, decode, execute แต่ละ instruction ของโปรแกรม

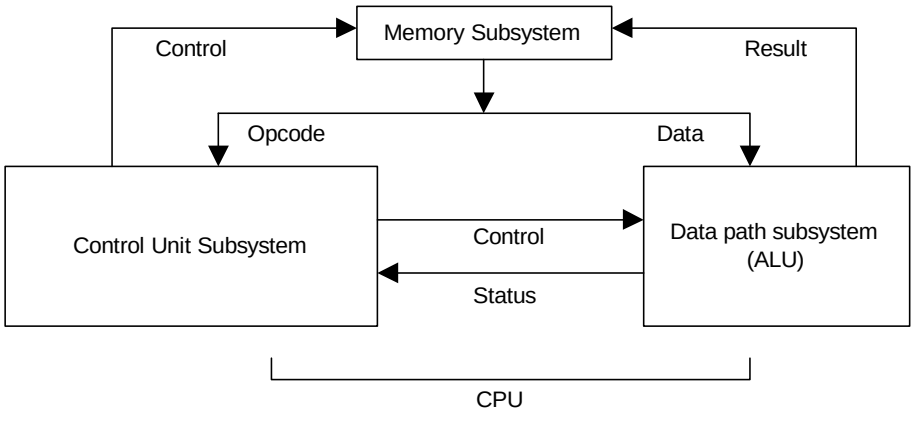
ตารางที่ 1 Computer instruction repertoire

Opcode	Mnemonic	Register Transfer	Description of Instruction
000	CLA	A ← 0	Clear A
001	CMP	A ← A’	Complement A
010	INC	A ← A+1	Increment A
011	NEG	A ← A’+1	Negate A = 2s complement A
100	LDA	A ← B	Transfer B to A
101	ADD	A ← A+B	Add B to A
110	SUB	A ← A + B’ + 1	Subtract B from A
111	STA	Z ← A	Pass A

step 2 : Conceptualization :

a. System structure diagram : เขียน Block Diagram ของโครงสร้างของระบบซึ่งแบ่งระบบออกเป็น

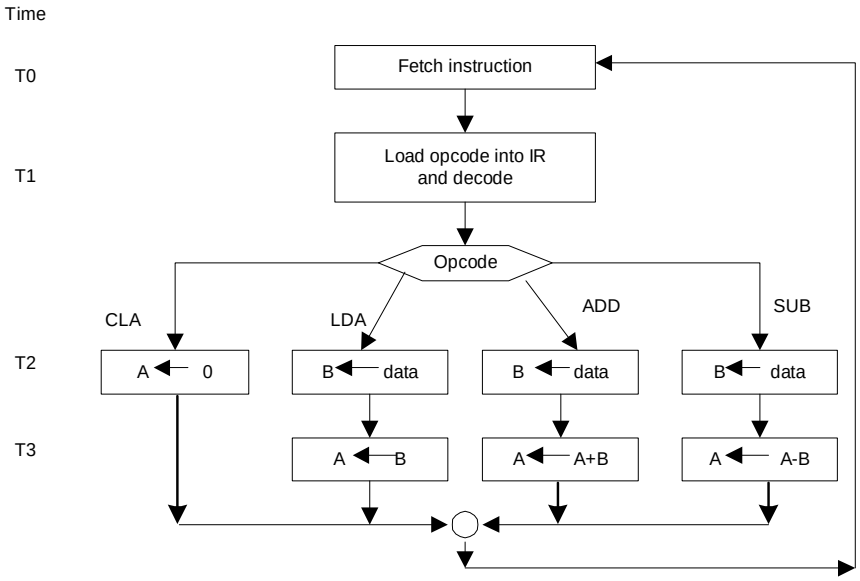
1. data path subsystem
2. control unit subsystem ที่ประกอบด้วย NS generator และ Output decoder
3. memory subsystem



รูปที่ 4 Structure Diagram ของ Processor ของ Computer

b. Control Flow Diagram : บางส่วนของ Control flow diagram (แสดงคำสั่ง CLA, LDA, ADD, SUB) แสดงดังรูปที่ 5 แต่ละ instruction cycle แบ่งเป็น 4 ช่วงเวลา คือ T0, T1, T2, T3

step 3 : Solution / Simplification
a. Data Path subsystem
b. Control unit subsystem
c. Memory subsystem



รูปที่ 5 control flow diagram (บางส่วน)

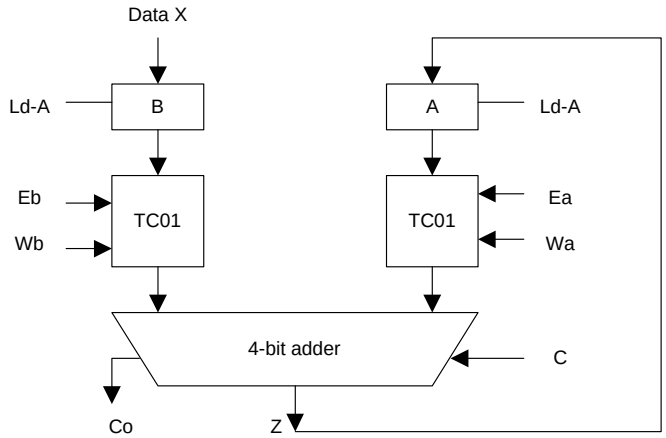
step 4 : Realization
a. สร้าง Data Path subsystem
b. สร้าง Control unit subsystem แล้วรวม Data Path กับ Control unit เป็น CPU
c. สร้าง Memory subsystem แล้วรวม memory กับ CPU เป็น processor

Design and Implementation of a data path subsystem

Data Path Subsystem อย่างง่ายสามารถสร้างขึ้นจากการรวม ALU เข้ากับรีจิสเตอร์แบบ data-storage 2 ตัว ดังต่อไปนี้

รีจิสเตอร์ B สำหรับเก็บค่า data X
Accumulator A สำหรับเก็บผลของการคำนวณทางคณิตศาสตร์และลอจิก

รีจิสเตอร์ A และ B จะถูก Load โดยการ strobe สัญญาณควบคุม Ld-A และ Ld-B จากนั้น O/P ของ Binary adder จะนำกลับไปเป็น I/P ของ Accumulator และ O/P ของ Accumulator ก็จะนำกลับมาเป็น I/P ของenable gate (AND) ของวงจร TC01 อีก ดังรูปที่ 6



รูปที่ 6 Block Diagram ของ data path subsystems

Data path Subsystem จะมีความสามารถในการ execute คำสั่งของ Computer ดังตารางที่ 2 ถ้าControls Eb, Wb, Ea, Wa และ C ได้ถูก set ค่าไว้อย่างเหมาะสม I/P ของ Adder จะคงค่าไว้ จนกว่า O/P ของ Adder จะถูก Load เข้าสู่ Accumulator A

ตารางที่ 2 Instruction Execution Table

Instruction Mnemonic	Controls Eb	Wb	Ea	Wa	C	Register Transfer T2	T3
CLA	0	0	0	0	0	A ← 0	
CMP	0	0	1	1	0	A ← A’	
INC	0	0	1	0	1	A ← A+1	
NEG	0	0	1	1	1	A ← A’+1	
LDA	1	0	0	0	0	B ← Data	A ← B
ADD	1	0	1	0	0	B ← Data	A ← A+B
SUB	1	1	1	0	1	B ← Data	A ← A-B
STA	0	0	1	0	0	Z ← A	

จะเห็นว่ามี 5 คำสั่งคือ CLA, CMP, INC, NEG, STA ที่ไม่ต้องการ Register และมี 3 คำสั่งที่ต้องการ คือ LDA, ADD, SUB นอกจากนี้บาง Instruction ต้องการ Microoperation ที่น้อยกว่า ก็จะใช้ช่วงเวลาที่น้อยกว่า

ตัวอย่างต่อไปนี้แสดงให้เห็นการทำงาน ของ data path subsystem เราจะทำการบวก list ของตัวเลข (0001, 0010, 0011, ….) เข้าไปโดยใช้ Algorithm ของ partial-sum ซึ่ง ลำดับของ Instruction จะถูก execute เพื่อทำตาม Algorithm ต่อไปนี้ :

1. ป้อนตัวเลขแรก (0001) เข้าที่ Input X และทำการ Strobe สัญญาณ Ld-B เพื่อ โหลด รีจิสเตอร์ B แล้ว Set สัญญาณควบคุม Eb, Wb, Ea, Wa และ C เป็น 1, 0, 0, 0 และ 0 ค่า Control จะทำให้วงจร adder ผ่านค่า B (= 0001) เข้าไปยัง Adder ผ่านออกไปยัง Output Lines จากนั้น เมื่อสัญญาณ Ld-A ถูก Strobe O/P ของ Adder จะถูกโหลด เข้าไปยัง Accumulator A
2. เพื่อเพิ่มตัวเลขถัดไปเข้าไปใน List ให้ป้อนตัวเลขนั้น (0010, 0011, …) เข้าไปยัง Input X แล้ว Strobe สัญญาณ Ld-B เพื่อ โหลด B แล้ว Set สัญญาณควบคุม Eb, Wb, Ea, Wa และ C เป็น 1, 0, 1, 0 และ 0 Strobe สัญญาณ Ld-A เพื่อโหลด ผลลัพธ์ A+B เข้าไปยัง A

3. ทำตาม Step 2 คือการป้อนตัวเลขใน List เข้าไปยัง Input X ค่า Control จะยังคงเป็น 1, 0, 1, 0 และ 0 เพื่อบวก A และ B หลังจากทำตัวเลขสุดท้ายใน list แล้ว ใน Accumulator A จะเป็นผลบวกของตัวเลขใน List ทั้งหมด

Computer แบบ Manual Control นี้จะเป็นอุปกรณ์สำหรับการคำนวณที่สามารถทำ Operation ทางคณิตศาสตร์และลอจิก ซึ่งขึ้นอยู่กับการ Set ค่า Control ในแบบ Manual data path subsystem นี้จึงสามารถเรียกได้ว่าเป็น Computer แบบ Manual Control ซึ่งมี Instruction Set ดังตารางที่ 1

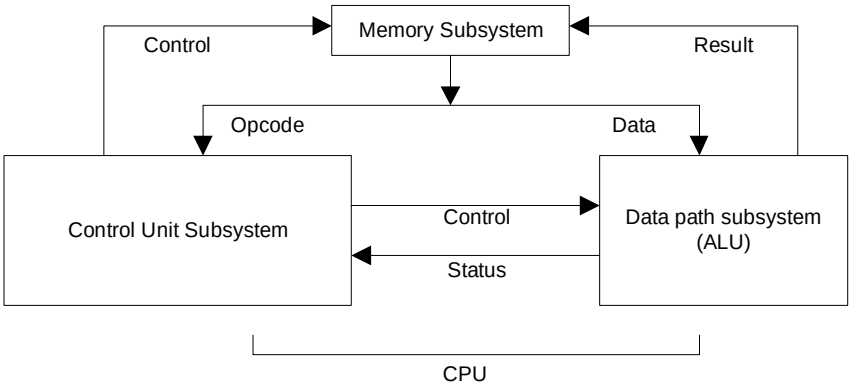
Design and Implementation of A control unit subsystem

Data Path Subsystem สามารถใช้เป็น Computer แบบ manual ที่มีความสามารถที่จะ execute คำสั่งต่าง ๆ ได้ แต่การทำงานของมันต้องมีคนคอยกำหนดค่าของ Eb, Wb, Ea, Wa และ C ของแต่ละ Instruction และ คอย strobe รีจิสเตอร์ ตลอดเวลา เพื่อเพิ่มประสิทธิภาพของมันให้เป็นแบบกึ่งอัตโนมัติ (Semiautomatic) จึงต้องมีการออกแบบส่วนของ Control Unit ซึ่งจะ

- Fetches (reads) คำสั่ง (instruction) ของโปรแกรมออกมาจาก Memory

- Decode แต่ละ instruction
- ส่งสัญญาณควบคุม ไปยัง data path ณ เวลาที่เหมาะสมเพื่อให้ทำงานตาม instruction

data path และ control unit สามารถนำมารวมกันเป็น CPU ของ processor ดังรูปที่ 7

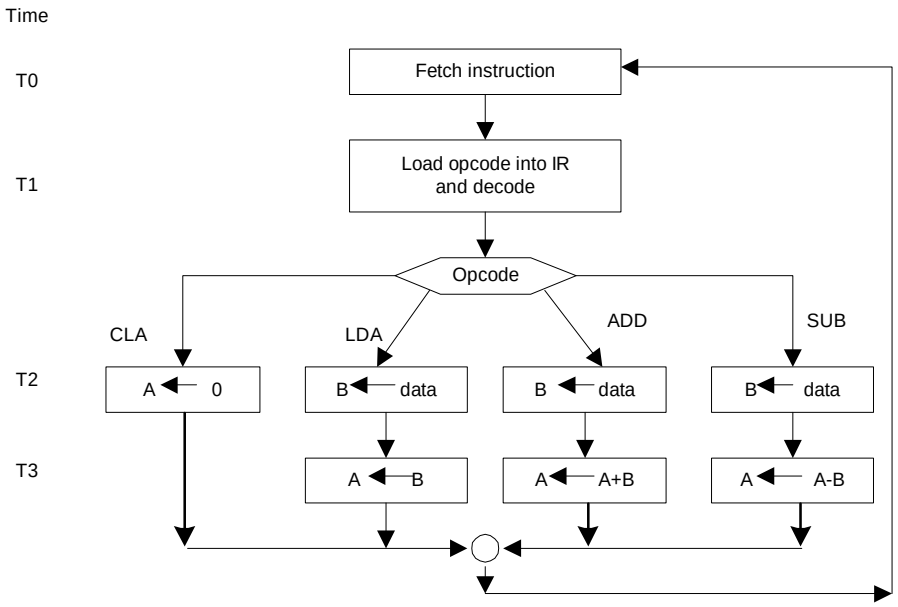


รูปที่ 7 Structure Diagram ของ computer processor

Design of Control Unit Subsystem

control unit subsystem ต้องทำการ fetch แต่ละ instruction จาก memory, จากนั้น decode คำสั่งให้เป็น opcode และสร้างสัญญาณไปควบคุมส่วนของ data path ณ เวลาที่เหมาะสม เพื่อให้มีการ execute คำสั่งนั้น ๆ

control algorithm จะทำการ fetch, decode, execute แต่ละ instruction ใน 4 ช่วงเวลา T0, T1, T2, T3 ดังรูปที่ 8



รูปที่ 8 Control flow diagram ของ Process control algorithm

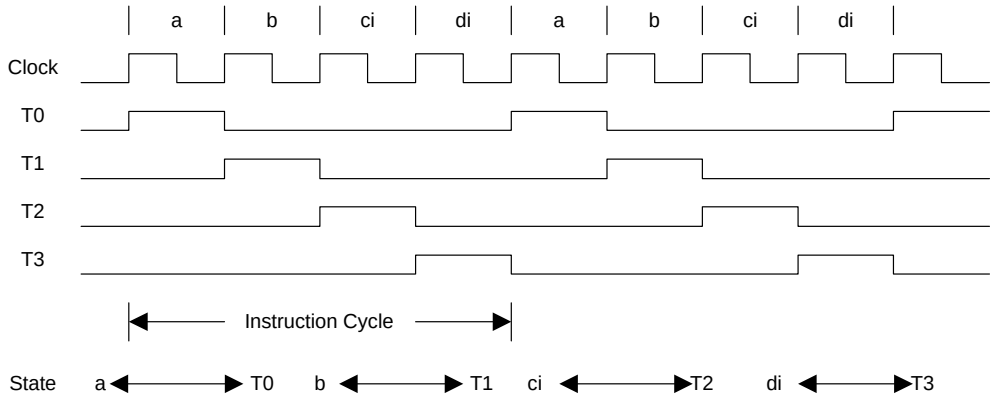
T0 instruction fetched (read from memory)

T1 opcode load เข้าไปยัง instruction register IR,
decode โดย 3:8 decoderได้ opcode

T2-T3 microoperations (register transfer) ทำตาม instruction นั้น ๆ

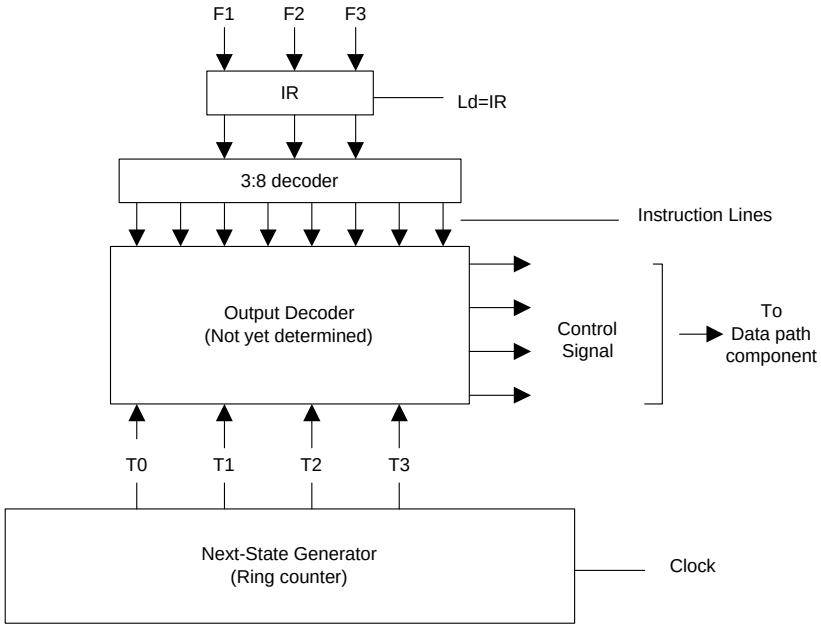
Control Unit Next-State Generator

Next State Generator ของ control unit subsystem จะสร้างขึ้นโดยใช้วงจร Ring counter โดยจะสร้าง พัลส์ T0, T1, T2, T3 ขึ้นมา Timing Diagram ของวงจร ring counter แสดงให้เห็นในรูปที่ 9



รูปที่ 9 Timing Diagram ของ 4-bit Ring counter

โครงสร้างของ control unit subsystem สามารถเขียนเป็น Block Diagram ดังรูปที่ 10



รูปที่ 10 Block diagram ของ control unit subsystem

Control Unit Output Decoder

Block Diagram ของ Control Unit แสดงให้เห็นว่า output decoder มี อินพุท 8 เส้น (จากการ Decode opcode) และ I/P 4 time pulse T0, T1, T2, T3

ใน State a, b, ci และ di output decoder จะสร้างสัญญาณ ควบคุม โดยใช้สัญญาณจาก instruction ที่ active กับสัญญาณ time pulse T0, T1, T2, T3 เป็นตัวกำหนด สัญญาณควบคุมเหล่า

นี้จะนำไปควบคุมส่วนของ Data Path เพื่อทำการ fetch, decode และ execute คำสั่ง Instruction-time event table ของคำสั่ง ต่าง ๆ เป็นดังตารางที่ 3

ตารางที่ 3 Instruction time event table

Instruction (Opcode) \ Event / Time	To	T1	T2	T3
CLA	Fetch inst.	Ld-IR	[00000], Ld-A	-
(000)	Fetch inst.	Ld-IR	[00110], Ld-A	-
CMP	Fetch Inst.	Ld-IR	[00101], Ld-A	-
(001)	Fetch Inst.	Ld-IR	[00111], Ld-A	-
INC	Fetch Inst.	Ld-IR	Ld-B	[10000], Ld-A
(010)	Fetch Inst.	Ld-IR	Ld-B	[10100], Ld-A
NEG	Fetch Inst.	Ld-IR	Ld-B	[11101], Ld-A
(011)	Fetch Inst.	Ld-IR	Ld-MW	WR
LDA				
(100)				
ADD				
(101)				
SUB				
(110)				
STA				
(111)				

เมื่อ [#####] แทนการ Set ค่า Control Eb, Wb, Ea, Wa และ C

- การ Strobe Ld-IR เป็นการโหลด Opcode เข้าสู่ IR
- การ Strobe Ld- B เป็นการโหลด Data Word เข้าสู่ B
- การ Strobe Ld-A เป็นการโหลด ผลลัพธ์ เข้าสู่ A
- การ Strobe Ld-MW เป็นการโหลด Data Word เข้าสู่ Memory
- WR เป็นสัญญาณ memory write

- Data Path Control Eb, Wb, Ea, Wa และ C จะเป็นสัญญาณที่ Active ณ เวลา Ti
- แต่ละ Product term จะเป็น logical product ของเวลา Ti และ instruction
- Logic Equation ของแต่ละ Control Point ของ Data Path สามารถหาได้โดยนำทุกเทอมที่เกี่ยวข้องกับ control point นั้นมารวมกัน ตัวอย่างเช่น Control Equation ของการโหลด Register A คือ

Ld-A = T2. CLA + T2. CMP + T2. INC + T2. NEG + T3. LDA + T3.ADD + T3.SUB + T2. STA

T0 Fetch Instruction

T1 Strobe Ld-IR เพื่อโหลด opcode ของคำสั่งเข้าสู่ register IR ซึ่ง O/P ของมัน ต่อเข้ากับ I/P ของวงจร 3:8 decoder (opcode จะถูก decode เป็นการ active คำสั่ง 1 คำสั่ง ไปยังส่วน O/P Decoder ของ control unit)

T2-T3 O/P Decoder จะสร้างสัญญาณไปควบคุมส่วนของ Data Path

การ Active ของ Control Point ใน Data Path ขึ้นอยู่กับ Instruction ที่กำลังถูก execute อยู่ ณ State ปัจจุบันของ Algorithm ตารางที่ 4 แสดงสัญญาณควบคุมต่างๆ ที่จะ Active ในแต่ละช่วงเวลา

ตารางที่ 4 P-term/control point activation table

Opcode . Time	Control Points									
	Fetch Ld-MW	Ld-IR WR	Ld-B	Eb	Wb	Ea	Wa	C	Ld-A
ALL . T0	1								
ALL . T1		1							
CLA . T2				0	0	0	0	0	1
CMP . T2				0	0	1	1	0	1
INC . T2				0	0	1	0	1	1
NEG . T2				0	0	1	1	1	1
LDA . T2			1						
LDA . T3				1	0	0	0	0	1
ADD . T2			1						
ADD . T3				1	0	1	0	0	1
SUB . T2			1						
SUB . T3				1	1	1	0	1	1
STA . T2									
STA . T3	1								
	1								

คอลัมน์ของ opcode และ time จะเป็นการ AND ทุกแถวของ control point จะเป็น OR จะได้ logic equation ดังต่อไปนี้

Fetch = T0
Ld-IR = T1
Ld-B = T2 . LDA + T2 . ADD + T2 . SUB
Eb = T3 . LDA + T3 . ADD + T3 . SUB

Wb = T3 . SUB
Ea = T2 . CMP + T2 . INC + T2 . NEG + T3 . ADD + T3 . SUB
Wa = T2 . CMP + T2 . NEG
C = T2 . INC + T2 . NEG + T3 . SUB
Ld-A = T2. CLA + T2. CMP + T2. INC + T2. NEG + T3. LDA + T3.ADD + T3.SUB

Control Point เหล่านี้จะ Active ที่กึ่งกลางของ Ti คือที่ขอบขาขึ้นของสัญญาณ eclk

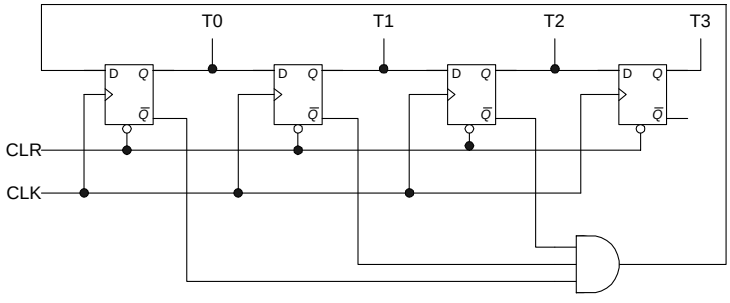
Ld-IR = T1 . eclk
Ld-B = (T2 . LDA + T2 . ADD + T2 . SUB) . eclk
Ld-A = (T2. CLA + T2. CMP + T2. INC + T2. NEG + T3. LDA + T3.ADD + T3.SUB) . eclk

ส่วน control point ที่เหลือจะ active ด้วยระดับของมันเอง

Fetch = T0
Eb = T3 . LDA + T3 . ADD + T3 . SUB
Wb = T3 . SUB
Ea = T2 . CMP + T2 . INC + T2 . NEG + T3 . ADD + T3 . SUB
Wa = T2 . CMP + T2 . NEG
C = T2 . INC + T2 . NEG + T3 . SUB

Implementation of an Clock-Ring Counter Circuit

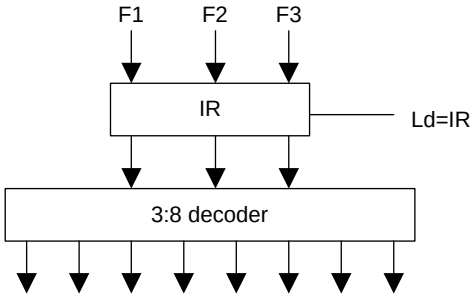
Ring counter ที่จะสร้างสัญญาณ pulse 4 ลูก ที่จะใช้ต่อกับแต่ละ opcode เพื่อให้ control unit สร้างสัญญาณควบคุมที่ต้องการสำหรับการ active ส่วนต่างๆใน (IR, A, A-enable, B-enable, etc.) มี Logic Diagram ดังรูปที่ 11



รูปที่ 11 Logic Diagram ของวงจร Ring-Counter

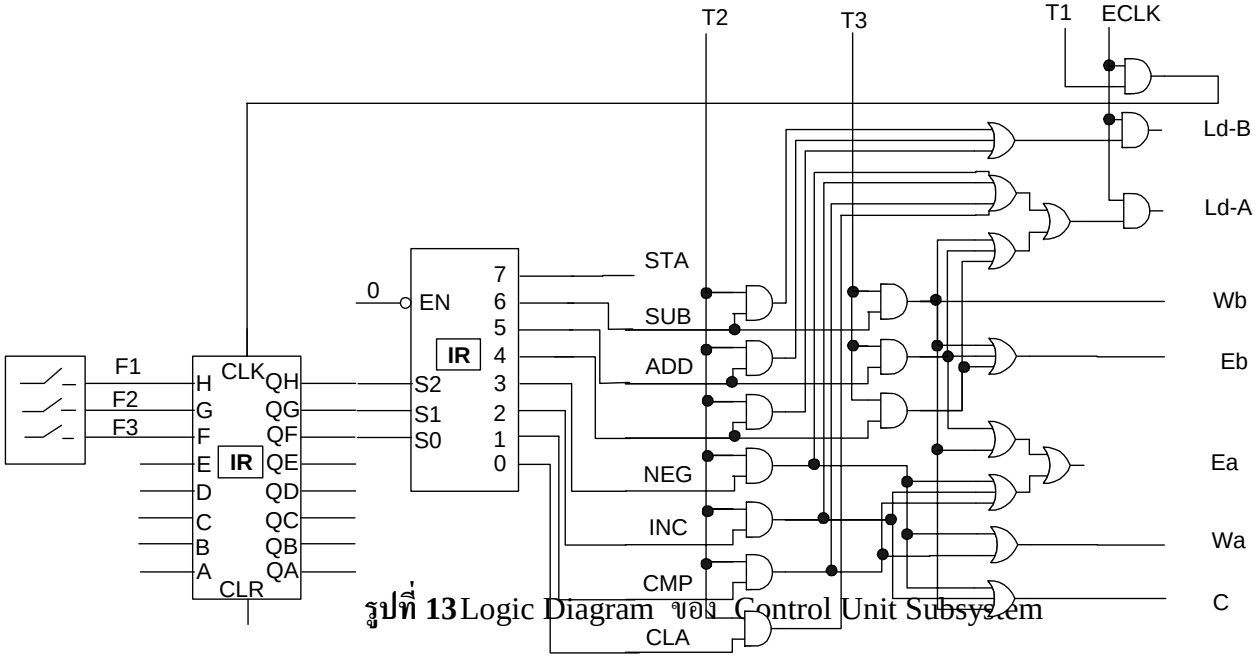
Implementation of an Instruction Register and Decoder

ให้ F1, F2, F3 เป็น opcode ขนาด 3 บิตของ instruction ที่กำลังถูก execute มันจะถูก load เข้าไปยัง IR และถูก decode โดยวงจร 3:8 decoder ซึ่งแต่ละ O/P ของ decoder จะเป็นแต่ละ instruction ใน instruction set รีจิสเตอร์ IR และวงจร decoder ที่แสดงใน block diagram ของ control unit ในรูปที่ 10 จะสร้างขึ้นโดยใช้ รีจิสเตอร์ แบบ edge-triggered ขนาด 4 บิต และวงจร 3:8 decoder โดย O/P ของ decoder นั้น Active low



รูปที่ 12 Instruction Register and Decoder

Implementation of Control Unit

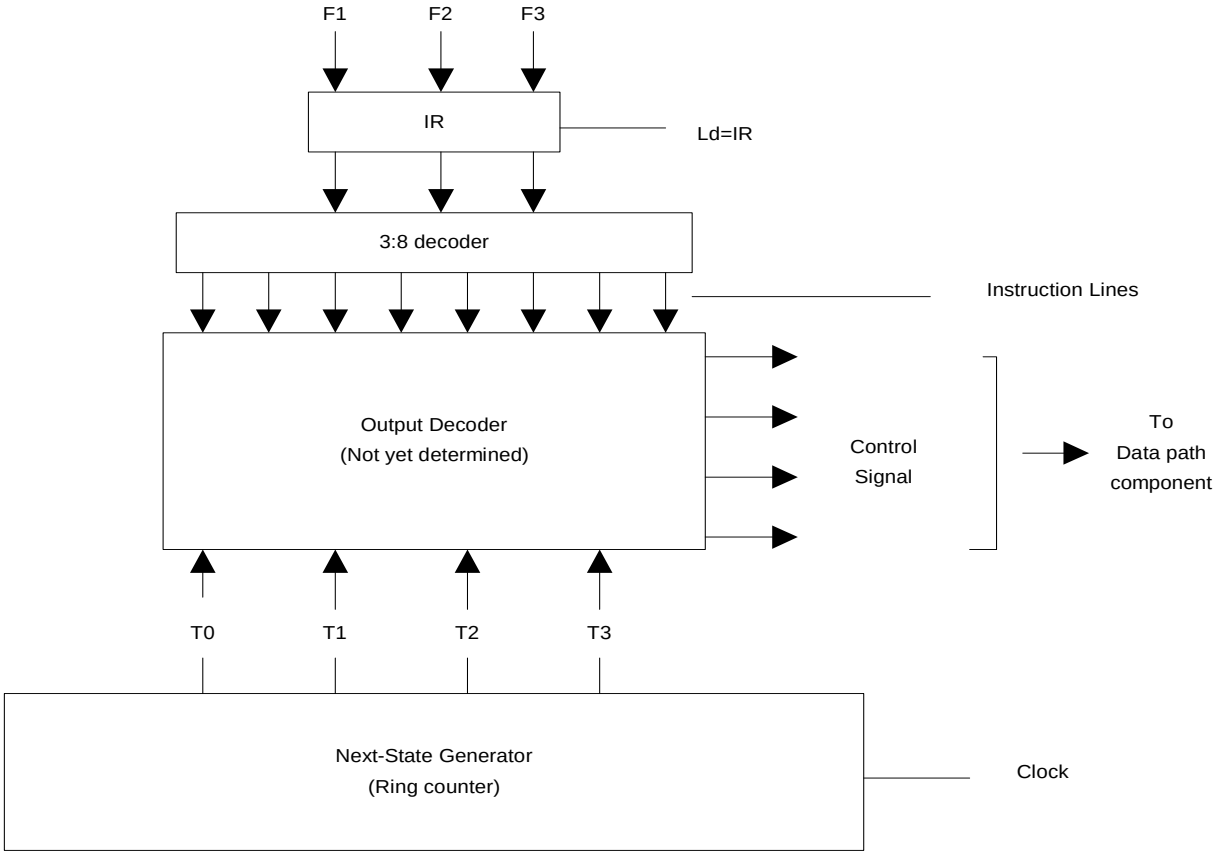


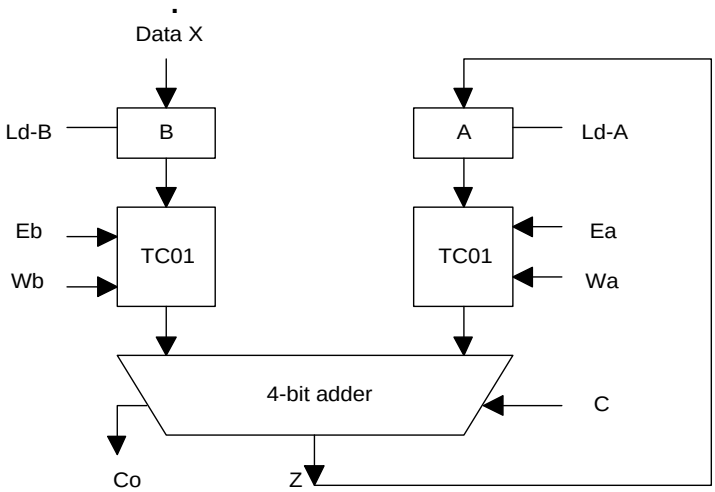
รูปที่ 13 Logic Diagram ของ Control Unit Subsystem

control unit subsystem สามารถสร้างขึ้นได้ดัง Logic Diagram รูปที่ 13

Integration of Control Unit and Data Path to Form a CPU

การสร้าง CPU ทำได้โดยรวม control unit และ data path subsystems เข้าด้วยกัน ดังรูปที่ 14





รูปที่ 14 Block Diagram ของ CPU