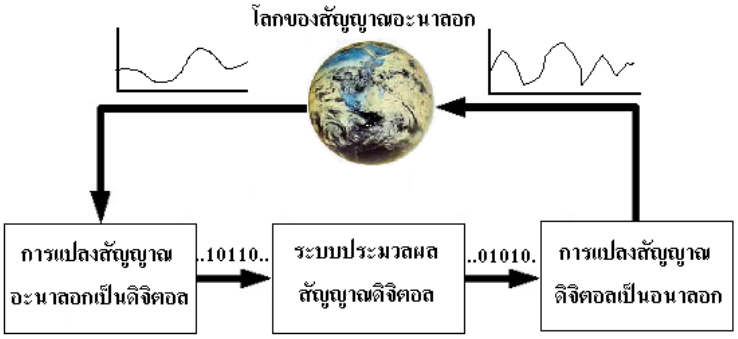
16. การเชื่อมต่อกับสัญญาณอนาลอก

14.1 บทนำ

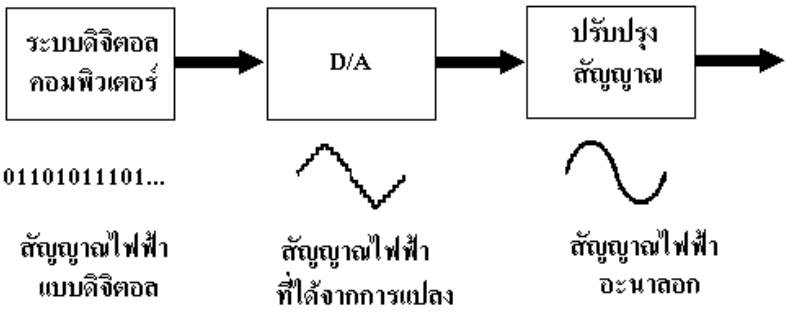


รูปที่ 14.1 แสดงขบวนการแปลงสัญญาณระหว่าอะนาลอกกับดิจิตอล

โดยทั่วๆไปสัญญาณที่ปรากฏอยู่ในทางธรรมชาติจะเป็นสัญญาณที่มีความต่อเนื่อง และมีขนาดได้หลายขนาดไม่จำกัด สัญญาณลักษณะนี้เรียกว่า “สัญญาณอนาลอก” ถ้าต้องการนำสัญญาณอะนาลอกนี้มาประมวลผลด้วยระบบดิจิตอล จะต้องมีการเปลี่ยนสัญญาณนี้ให้เป็นสัญญาณดิจิตอลเสียก่อน ระบบนี้เรียกว่า การแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล (Analog to Digital conversion หรือ A/D) และในทางตรงข้ามกัน เมื่อประมวลผลได้แล้วสัญญาณที่ได้ยังเป็นสัญญาณดิจิตอลอยู่ ถ้าต้องการนำออกสู่โลกภายนอกแบบสัญญาณอะนาลอก ก็ต้องทำการแปลงสัญญาณให้เป็นสัญญาณอะนาลอกด้วยระบบแปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอก (Digital to Analog Conversion หรือ D/A)

14.2 วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอก (Digital-to-Analog Converter หรือ D/A)

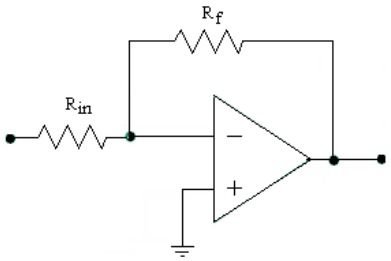
หมายถึงวงจรที่เปลี่ยนข้อมูลทางดิจิตอลให้เป็นค่าอนาลอก ค่าอนาลอกนี้อาจเป็นกระแสไฟฟ้าหรือแรงดันไฟฟ้าก็ได้



รูปที่ 14.2 แสดงขบวนการแปลงสัญญาณจากดิจิตอลเป็นอะนาลอก

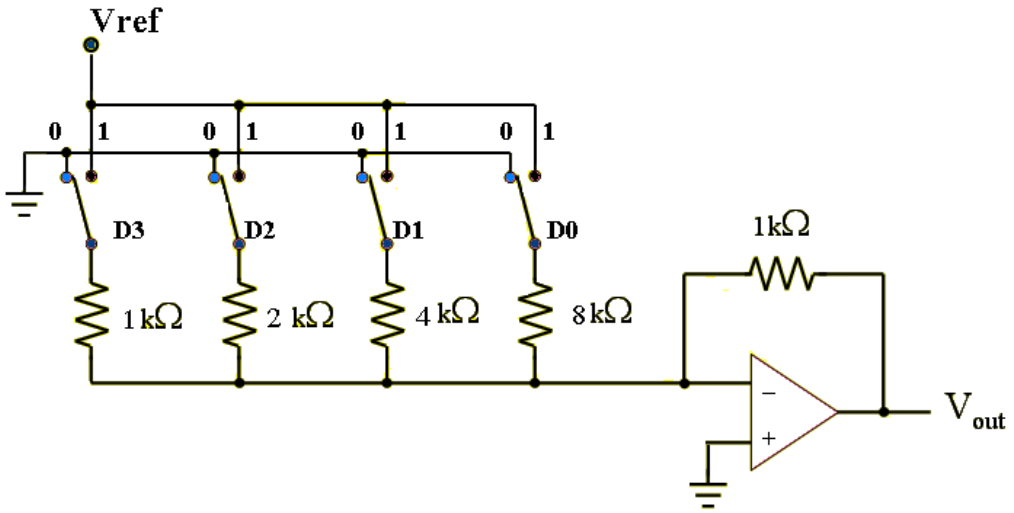
เทคนิคการแปลงสัญญาณดิจิตอลเป็นสัญญาณอะนาลอกมีหลายวิธีเช่น Binary Weighted DAC และ R/2R Ladder การทำงานของแต่ละวิธีมีรายละเอียดดังนี้

14.2.1 Binary Weighted DAC



รูปที่ 14.3 แสดงวงจรออปแอมป์แบบกลับสัญญาณ (Inverting Amp.)

คุณสมบัติของวงจรประเภทนี้ได้แก่ ความต้านทานอินพุทสูง อัตราขยายแรงดันมีค่าเท่ากับ $-R_f/R_{in}$



รูปที่ 14.4 วงจร Binary Weighted DAC

ตามรูปที่ 14.4 จะเห็นได้ว่าแรงดันเอาท์พุทต่ำสุดเกิดเมื่อสวิตซ์ D3- D0 อยู่ตำแหน่ง ‘0’

Vout = 0 โวลท์

ถ้า D3- D0 อยู่ตำแหน่ง ‘1’ จะได้แรงดันเอาท์พุทสูงสุด

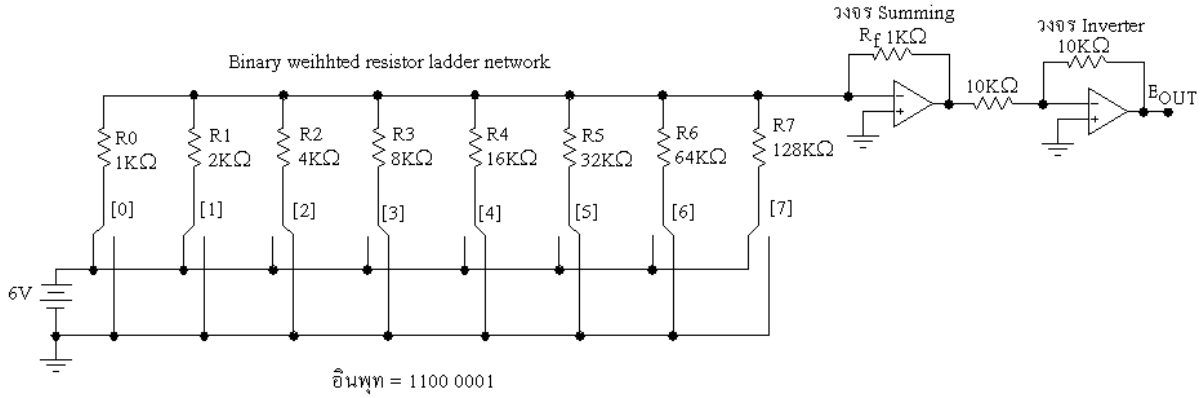
Vout = - Vref x Rf x(1/1k + 1/2k + 1/4k + 1/8k)

Vout = -Vref x 1 x(1/1 + 1/2 + 1/4 + 1/8)

Vout = -Vref x (1 + .5 + .25 + .125)

Vout = -1.875 x Vref

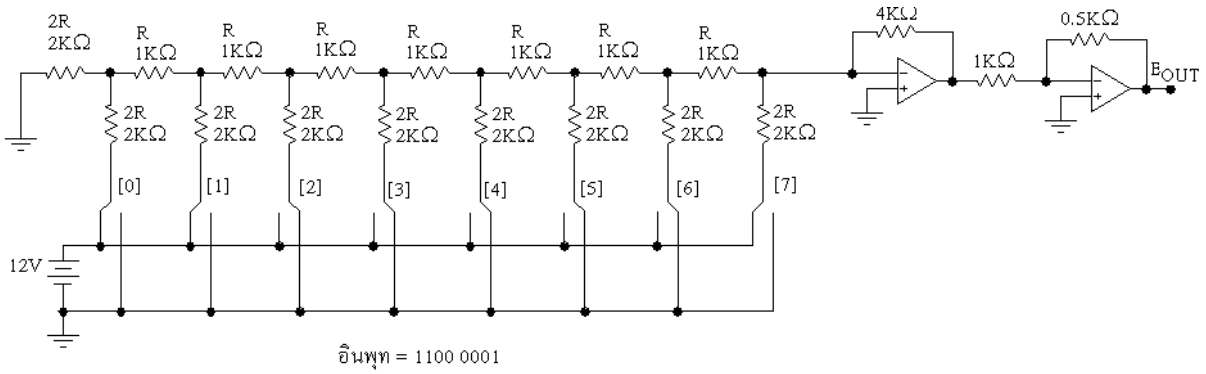
เช่นถ้า Vref = +5V Vout เมื่ออินพุทเป็น ‘1’ หมดจะได้ -9.375 โวลท์ แต่การทำวงจรให้มีจำนวนบิตมากขึ้นต้องใช้ความต้านทานขนาดใหญ่ขึ้น ตัวอย่าง จงคำนวณหาค่า Eout



รูปที่ 14.5 ตัวอย่างวงจร DAC ขนาด 8 บิต

14.2.2 R/2R Ladder

เป็นวิธีที่ใช้กันทั่วๆไปในการผลิตเป็นวงจรรวม เพราะว่า ใช้ค่าความต้านทานต่ำและใช้เพียง 2 ค่าคือ 1R กับ 2R เช่น 1 K กับ 2K ตัวอย่างวงจรมีลักษณะตามรูปที่ 14.6 ซึ่งสามารถวิเคราะห์วงจรโดยใช้ Thevenin’s theorem



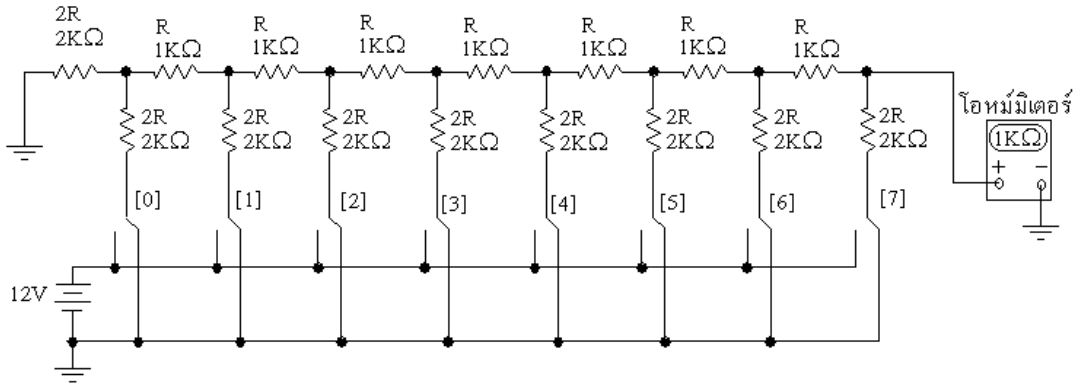
รูปที่ 14.6 ตัวอย่างวงจร DAC แบบ R/2R Ladder ขนาด 8 บิต

การวิเคราะห์วงจรโดยใช้ทฤษฎี Thevenin

หา Rth ของ Thevenin

ที่ตำแหน่งบิตต่างๆ สามารถหา Rth ได้ดังนี้

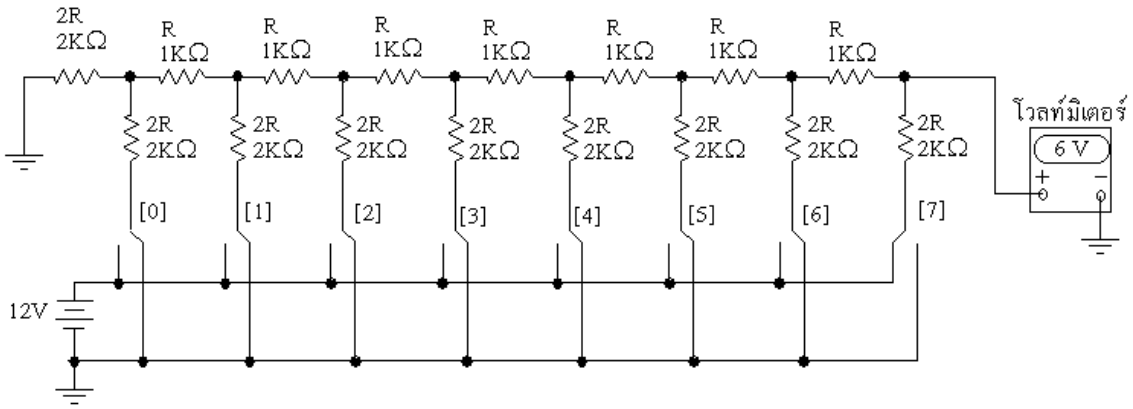
บิต 0 Rth = 1 k, บิต 1 Rth = 1 k,บิต 2 Rth = 1 k,บิต 3 Rth = 1 k,บิต 4 Rth = 1 k,บิต 5 Rth = 1 k, บิต 6 Rth = 1 k และ บิต 7 Rth = 1 k



รูปที่ 14.7 วงจร Thevenin equivalent เพื่อหา Rth

หา Vth ของ Thevenin

ที่ตำแหน่งบิตต่างๆ สามารถหา Vth ได้ดังนี้



รูปที่ 14.8 วงจร Thevenin equivalent เพื่อหา Vth เมื่อบิต 7 เป็น '1'

บิต 0 Vth = 46.875 mV, บิต 1 Vth = 93.75 mV, บิต 2 Vth = 0.1875 V, บิต 3 Vth = 0.375 V,
บิต 4 Vth = 0.75 V, บิต 5 Vth = 1.5 V, บิต 6 Vth = 3V และ บิต 7 Vth = 6V

14.2.3 เทอมที่เกี่ยวกับ DAC

Resolution

เป็นขนาดที่เล็กที่สุดที่ D/A สามารถสร้างได้ ขนาดนี้ขึ้นอยู่กับจำนวนบิตของ D/A หรือคำนวณได้จาก

$$V_{res} = \left(\frac{V_{REF}}{2^n} \right)$$

จำนวนระดับของสัญญาณอะนาลอก

$$\text{Analog Levels} = 2^n$$

ค่าความเที่ยงตรง(Accuracy)

เป็นการเปรียบเทียบค่าจริงกับค่าที่คำนวณได้ ปกติจะกำหนดเป็นเปอร์เซนต์ของค่าเอาท์พุทเมื่อเต็มสเกล (Full scale)

ความเร็ว (Speed)

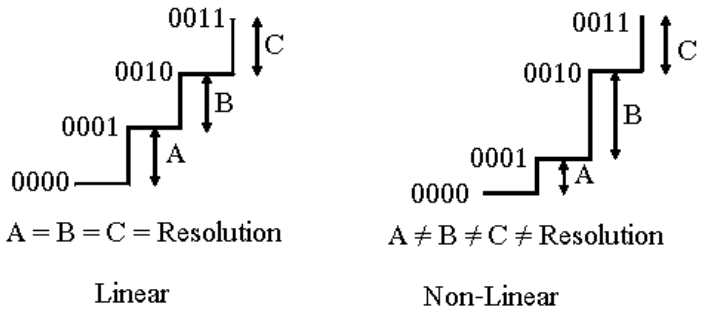
หมายถึงความเร็วของ Output settling time ซึ่งเป็นเวลาที่ต้องใช้เพื่อสร้างค่าเอาท์พุทเมื่ออินพุทมีการเปลี่ยนแปลง

การผิดพลาดใน DAC

DAC ที่ดีต้องให้ค่าเอาท์พุทออกมาตรงตามค่าอินพุท เมื่ออินพุทให้ค่ามากขึ้น สัญญาณเอาท์พุทก็ต้องมีค่ามากขึ้น และการเพิ่มขึ้นของอินพุทแต่ละขั้นก็ต้องให้ค่าเอาท์พุทเพิ่มขึ้นในแต่ละขั้นเท่าๆกันด้วย แต่ถึงกระนั้น ในทางเป็นจริง DAC ก็มีการผิดเพี้ยนเช่นกัน การผิดเพี้ยนใน DAC มี 2 แบบใหญ่ๆ คือ Non-linear distortion และ Non-monotonic distortion

Non-Linear Distortion:

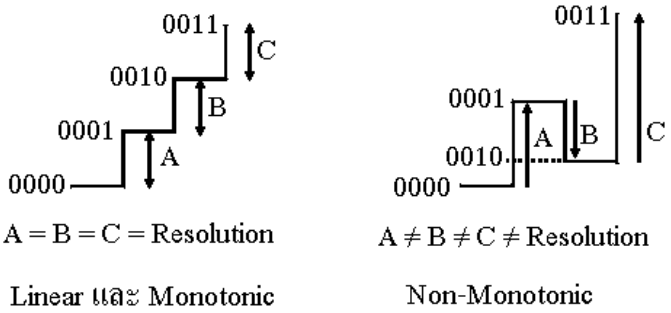
ลักษณะการผิดเพี้ยนแบบนี้จะให้ค่าเอาท์พุทของแต่ละสเต็ปไม่เท่ากัน



รูปที่ 14.9 การผิดเพี้ยนแบบ Non-linear

Non-Monotonic Distortion

ลักษณะการผิดเพี้ยนแบบนี้จะให้ค่าเอาท์พุทของแต่ละสเต็ปไม่เท่ากัน และบางครั้งให้ค่าเอาท์พุทลดลงทั้งๆที่อินพุทมีค่าเพิ่มขึ้น



รูปที่ 14.10 การผิดพลาดแบบ Monotonic

ตัวอย่าง DAC ขนาด 2 บิต มีแรงดันอ้างอิง 8 โวลท์และค่าความเที่ยงตรง @ ± 0.2% จงหาค่า resolution และค่าความเที่ยงตรงในเทอมของแรงดัน

Resolution = $1/2^2 = 1/4 = 25\%$ หรือ Resolution = $(1/4)(8V) = 2V$

Accuracy = $(\pm 0.2\%)(8V) = \pm 16mV$

สรุปการคำนวณ

$$V_{OUT} = V_{ref}^{-} + X_{10}\left(\frac{V_{ref}^{+} - V_{ref}^{-}}{2^n}\right)$$

$$V_{OUT} = V_{\min} + X_{10}\left(\frac{V_{\max} - V_{\min}}{2^n - 1}\right)$$

$$\text{Resolution} = \left(\frac{V_{ref}^{+} - V_{ref}^{-}}{2^n}\right)$$

$$\text{Resolution} = \left(\frac{V_{\max} - V_{\min}}{2^n - 1}\right)$$

$$V_{OUT} = V_{ref}^{-} + X_{10}\,\text{Resolution}$$

$$V_{OUT} = V_{\min} + X_{10}\,\text{Resolution}$$

V_{OUT} = Output Voltage

V_{ref}^{-} = Voltage Reference -

V_{ref}^{+} = Voltage Reference +

$V_{\min}$ = Minimum output voltage

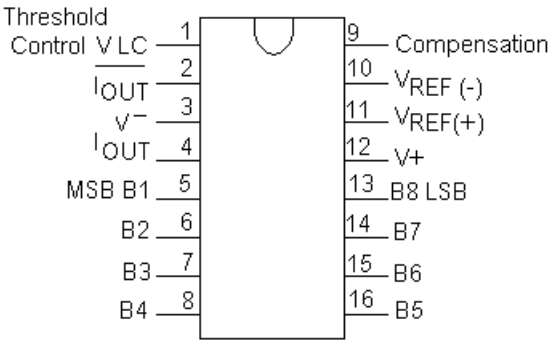
$V_{\max}$ = Maximum output voltage

X_{10} = Input value base 10

n = จำนวนบิต

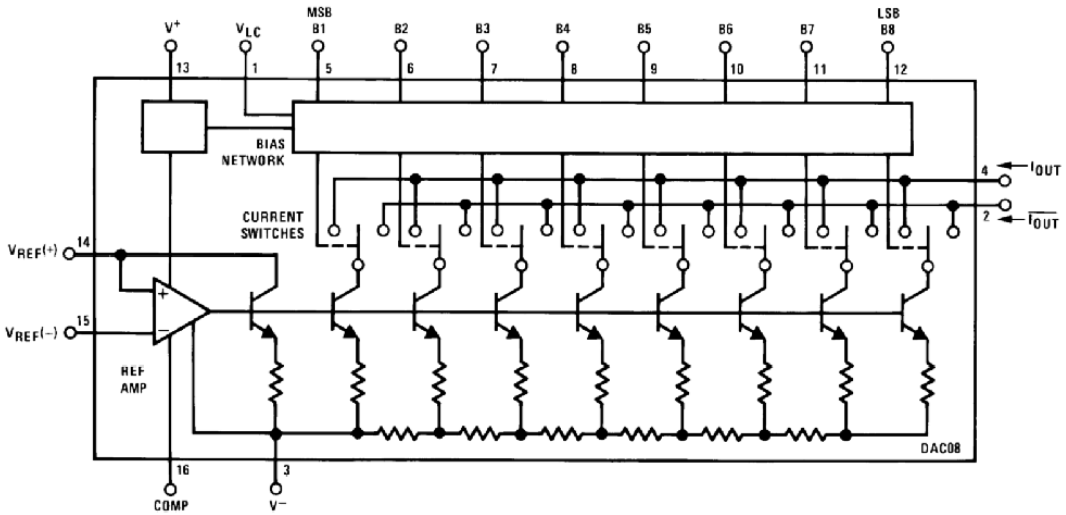
14.2.4 DAC0800

DAC 0800 เป็นวงจรร่วมที่ทำหน้าที่เป็น DAC มีอินพุท 8 บิต ให้เอาท์พุทเป็นกระแส ตำแหน่งขาสัญญาณเป็นตามรูป 4.11 และ บล็อกไดอะแกรมแสดงอยู่ในรูปที่ 14.12



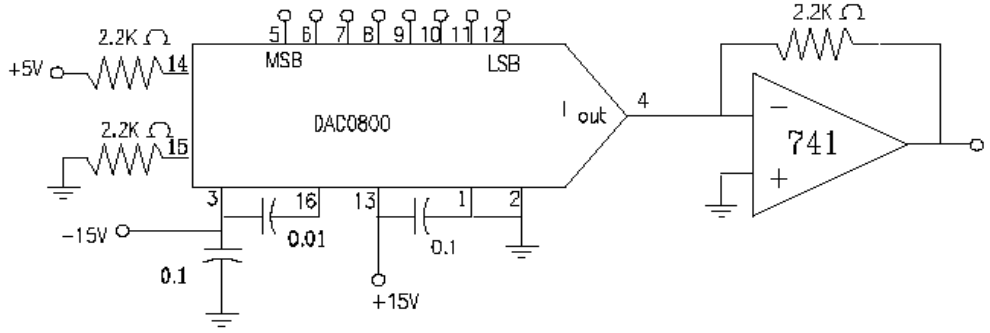
รูปที่ 14.11 ลักษณะตัวถังและขาสัญญาณของ DAC0800

ขาสัญญาณ	ชื่อ	ความหมาย
1	GND	Ground
2	IOUT’	Output
4.	IOUT	Output’
5	B1 → MSB	Input
	: : :	
12	B8 → LSB	Input
14	VREF(+)	Reference voltage for output
15	VREF(-)	Reference voltage for output



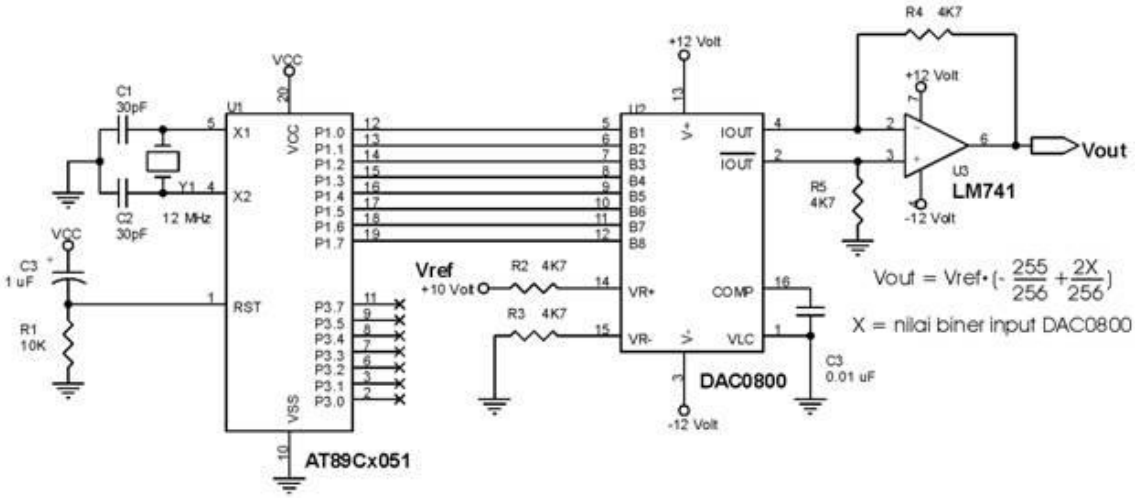
รูปที่ 14.12 ไดอะแกรมของ DAC0800

ตัวอย่างการต่อวงจรเพื่อให้เอาท์พุทเป็นแรงดัน



รูปที่ 14.13 การใช้งาน DAC0800

ตัวอย่างการต่อ AT89C2051 กับ DAC0800

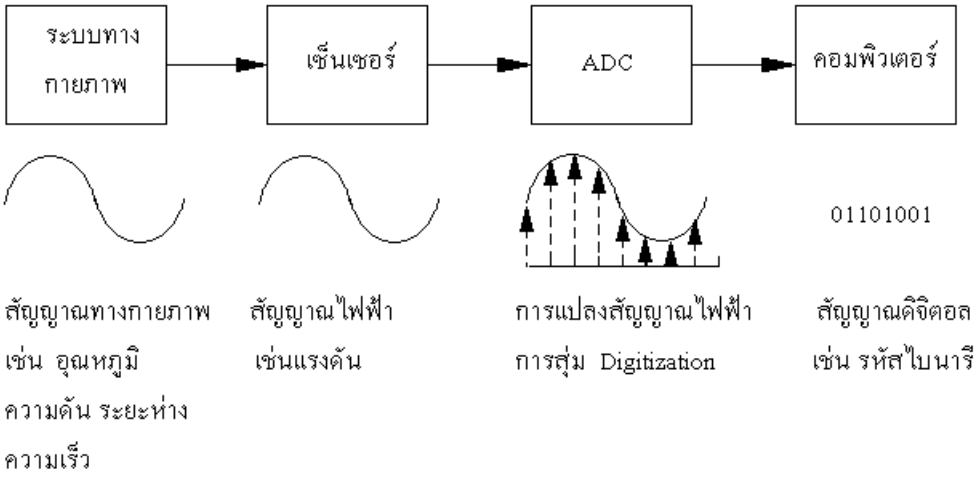


รูปที่ 14.14 การใช้งาน DAC0800 กับ AT89C2051

14.3 วงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล (Analog-to-Digital Converter หรือ A/D หรือ A2D)

สัญญาณในทางธรรมชาติทั่วๆไปเป็นสัญญาณที่มีความต่อเนื่องมีขนาดได้ไม่จำกัด การที่จะนำสัญญาณเหล่านี้มาประมวลผลด้วยระบบดิจิตอลคอมพิวเตอร์โดยตรงนั้นย่อมไม่สามารถทำได้ อีกประการหนึ่งสัญญาณเหล่านี้บางครั้งก็ไม่ได้อยู่ในรูปของสัญญาณไฟฟ้า ดังนั้นการนำมาประมวลผลด้วยระบบดิจิตอลต้องมีกระบวนการแปลงเสียก่อน ตามรูปที่ 14.15 แสดงถึงขั้นตอนการนำสัญญาณในทางธรรมชาติมาใช้งานในระบบดิจิตอล เริ่มต้นสัญญาณในทางธรรมชาติเช่นอุณหภูมิซึ่งเป็นพลังงานความร้อนต้องถูกแปลงให้เป็นพลังงานไฟฟ้าเสียก่อนด้วยด้วยอุปกรณ์ที่เรียกว่าเซ็นเซอร์ หรือทรานสดิวเซอร์ สัญญาณ

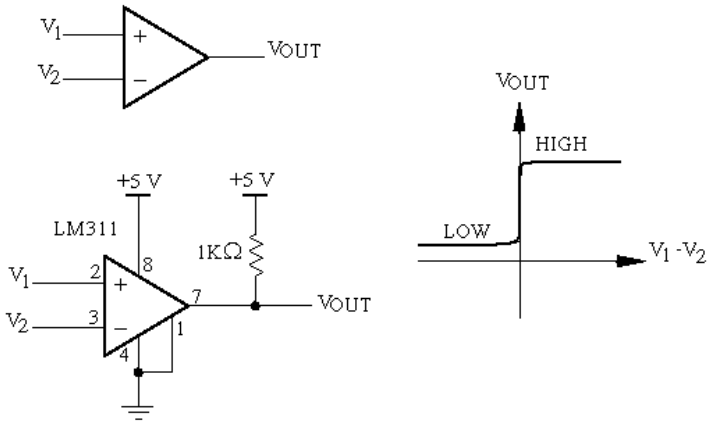
เอาท์พุทที่ได้จากเซ็นเซอร์นี้ยังคงเป็นสัญญาณที่มีความต่อเนื่อง หรือที่เรียกกันทั่วๆไปว่าสัญญาณอนาลอก สัญญาณนี้ต้องถูกแปลงให้เป็นสัญญาณดิจิตอลด้วยวงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล หรือที่เรียกว่า ADC หลังจากนี้จึงได้เป็นสัญญาไฟฟ้าที่อยู่ในรูปของสัญญาณดิจิตอลที่พร้อมจะนำไปประมวลผลในระบบดิจิตอลได้



รูปที่ 14.15 ลักษณะการใช้งาน ADC

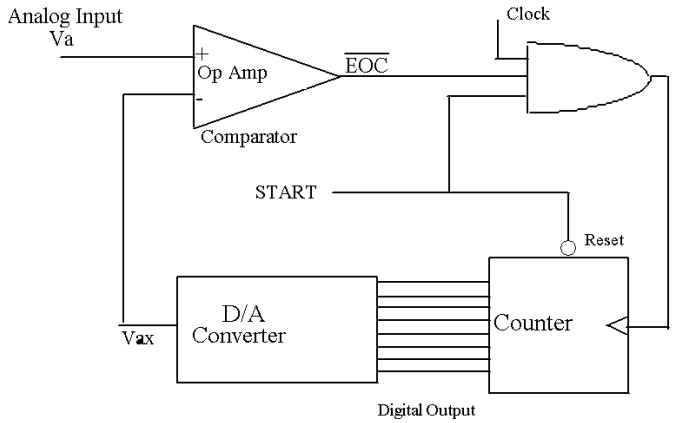
การแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลนั้นมีอยู่หลายวิธี เช่นตัวอย่างวงจรแปลสัญญาณอนาลอกเป็นสัญญาณดิจิตอลแบบง่ายๆ โดยการใช้วงจรAnalog comparator ตามรูปที่ 14.16 ซึ่งให้เอาท์พุทออกมาขนาด 1 บิต นอกจากนี้ยังมีวิธีอื่นๆอีกเช่น

- Digital Ramp ADC (Counter method)
- Successive Approximation
- Parallel Comparator (“Flash”)
- Dual Slope



รูปที่ 14.16 ลักษณะของ ADC ขนาด 1 บิต

14.3.1 Digital Ramp ADC (Counter method)

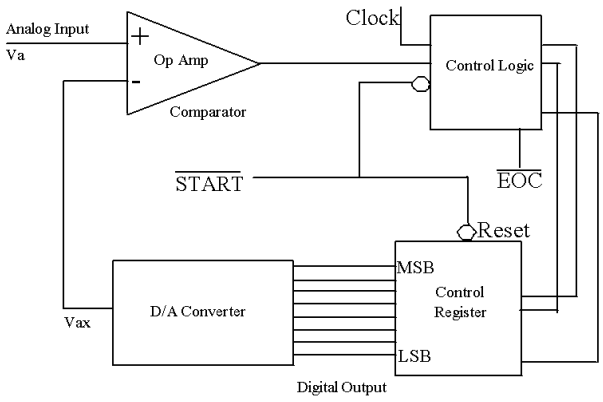


รูปที่ 14.17 ลักษณะของ ADC แบบ Counter methgod

บล็อกไดอะแกรมของวิธีนี้แสดงอยู่ในรูปที่ 14.17 โดยมีการทำงานดังนี้

1) เมื่อเริ่มสั่งให้แปลงสัญญาณ START = 0 จะเป็นการรีเซ็ตวงจรนับ (Counter) ให้ได้เป็น 0 ทุก บิต หลังจากนี้ START จะต้องเป็น 1 เพื่อเริ่มขบวนการทำงาน
2) ค่าจากวงจรนับจะถูกแปลงให้เป็นค่าอนาลอก Vax โดย DAC
3) ค่า Vax จะถูกนำไปเปรียบเทียบกับ ค่าแรงดันอินพุท Va โดยตัวเปรียบเทียบ (Comparator)
   - ถ้า Va > Vax เอาท์พุท $\overline{EOC}$ = 1 ทำให้ผลของการ AND ระหว่าง START $\overline{EOC}$ และ Clock ได้เป็นสัญญาณ Clock ป้อนเข้าวงจรนับ วงจรนับก็จะนับค่าขึ้นไปเรื่อยๆ ขบวนการก็จะเป็นไปตาม ขั้นที่ 2 และ 3 จนกว่า Va น้อยกว่าหรือเท่ากับ Vax
   - ถ้า Va < หรือ = Vax เอาท์พุท $\overline{EOC}$ = 0 วงจรนับจะหยุดนับ เป็นการเสร็จสิ้นการทำงาน

14.3.2 Successive Approximation



รูปที่ 14.18 ลักษณะของ ADC แบบ Successive Approximation

บล็อกไดอะแกรมของวิธีนี้แสดงอยู่ในรูปที่ 14.18 โดยมีการทำงานดังนี้

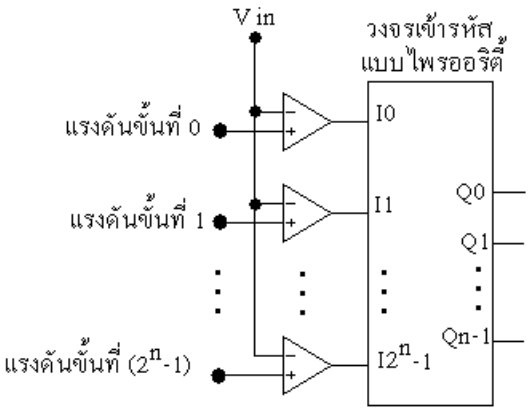
1) เมื่อเริ่มสั่งให้แปลงสัญญาณ START = 0 จะเป็นการรีเซ็ต Control logic และ Control Register ให้เป็น 0 ทุกบิต พร้อมทั้งกำหนดตำแหน่งบิตที่ต้องเซ็ตหรือรีเซ็ตเป็นบิตที่ 7 หลังจากนี้ START จะต้องเป็น 1 เพื่อเริ่มขบวนการทำงาน
2) ค่าจากวงจรนับจะถูกแปลงให้เป็นค่าอนาลอก Vax โดย DAC
3) ค่า Vax จะถูกนำไปเปรียบเทียบกับ ค่าแรงดันอินพุท Va โดยตัวเปรียบเทียบ (Comparator)
   - ถ้า Va > Vax เอาท์พุทของตัวเปรียบเทียบจะเป็น 1 และ $\overline{EOC}$ = 1 เมื่อสัญญาณ Clock เข้ามา Control logic จะเซ็ตบิตที่ระบุให้เป็น 1 พร้อมทั้งเลื่อนการระบุบิตลงมา 1 ตำแหน่ง แล้วกลับไปเปรียบเทียบใหม่
   - ถ้า Va < Vax เอาท์พุท $\overline{EOC}$ = 1 เมื่อสัญญาณ Clock เข้ามา Control logic จะรีเซ็ตบิตที่ระบุให้เป็น 0 พร้อมทั้งเลื่อนการระบุบิตลงมา 1 ตำแหน่ง แล้วกลับไปเปรียบเทียบใหม่
   - การทำงานจะวนอยู่ในข้อ 3 นี้จนกว่าจะทำครบทุกบิต (ในที่นี้มี 8 บิต) หรือ Va = Vax เอาท์พุท $\overline{EOC}$ จะเท่ากับ 0 วงจรจะหยุดทำงาน เป็นการเสร็จสิ้นการทำงาน

วิธีการแปลงแบบนี้ให้ความเร็วมากกว่าแบบ การนับ อีกทั้งราคาก็ถูกจึงเป็นวิธีที่นิยมใช้กันมาก

14.3.3 Parallel Comparator (“Flash") ADC

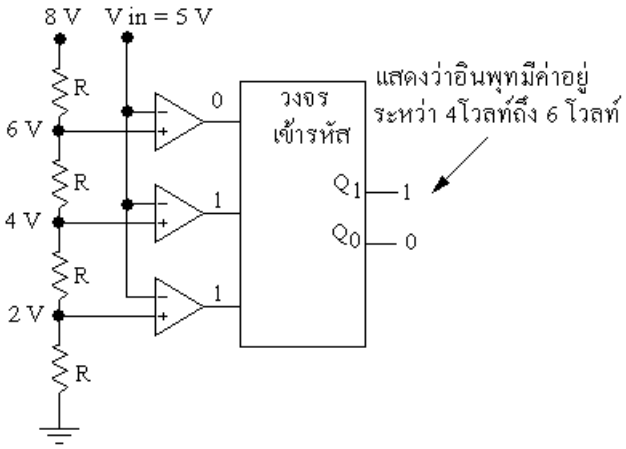
เป็นวิธีที่เร็วที่สุด ใช้ในสโคปแบบดิจิตอล สัญญาณวิดีโอ และการใช้งานที่ต้องการความเร็วสูง แต่ก็มีข้อเสียในด้านราคา เพราะว่าใช้ฮาร์ดแวร์ในการสร้างมาก ดังนั้นจึงเหมาะกับการใช้งานที่ต้องการจำนวนบิตน้อยๆ

ลักษณะของวงจรประกอบด้วยตัวเปรียบเทียบและวงจรเข้ารหัส จำนวนของตัวเปรียบเทียบเช่นถ้า



รูปที่ 14.19 ลักษณะของ ADC แบบ Flash

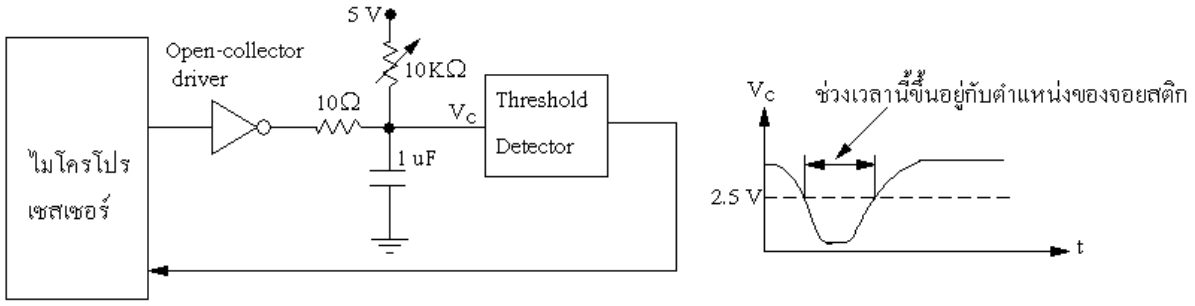
ต้องการ ADC ขนาด 2 บิตมีจำนวน 4 สเต็ป ซึ่งมีแรงดันอ้างอิง 8 โวลท์ ดังนั้นแต่ละสเต็ป 0 โวลท์ 2 โวลท์ 4 โวลท์ และ 6 โวลท์ ดังนั้นต้องใช้ตัวเปรียบเทียบ 3 ตัว ตามรูปที่ 14.20 ดังนั้นเมื่อป้อนแรงดันอินพุท เช่น 5 V เข้ามา ตัวเปรียบเทียบทั้ง 3 ตัวจะทำการเปรียบเทียบกับแรงดันอ้างอิงของแต่ละระดับได้เอาท์พุท เป็น 0 1 1 วงจรเข้ารหัสจึงให้เอาท์พุทเป็นรหัสไบนารี 10 ตามรูปที่ 14.20



รูปที่ 14.20 ตัวอย่าง ADC แบบ Flash ขาด 2 บิต

14.3.4 ADC แบบอื่นๆ

นอกจากแบบที่ได้กล่าวมาแล้ว ยังมีADC แบบอื่นๆอีกเช่น ADC ที่ใช้ใน จอยสติก (Joystick) ของคคอมพิวเตอร์ Apple II โดยมีรูปแบบตามรูปที่ 14.21 สำหรับ Threshold Detector อาจใช้ Op-amp ที่ใช้แรงดันอ้างอิงที่ขา inverting เป็น 2.5 โวลท์ หรือใช้ 74LS14 หรือ 74HCT14 ก็ได้ ส่วนตำแหน่งของจอยสติกกำหนดด้วย Potentiometer ซึ่งเป็นความต้านทานที่ปรับค่าได้ หลักการทำงานจะใช้การวัดค่า RC time constant ที่เกิดจากการประจุให้กับ C 1 μF ทั้งนี้การชาร์ตและดิสชาร์ต ประจุนี้ สั่งงานด้วยโปรแกรมของไมโครโปรเซสเซอร์



รูปที่ 14.21 ลักษณะของ ADC แบบ Apple II A-to-D for joystick

14.3.5 ข้อกำหนดของ ADC

Resolution

เหมือนกับใน DAC จะเป็นตัวกำหนดค่าความผิดพลาดของการ quantizing ซึ่งเป็นการผิดพลาดจากค่าที่แท้จริง

Accuracy

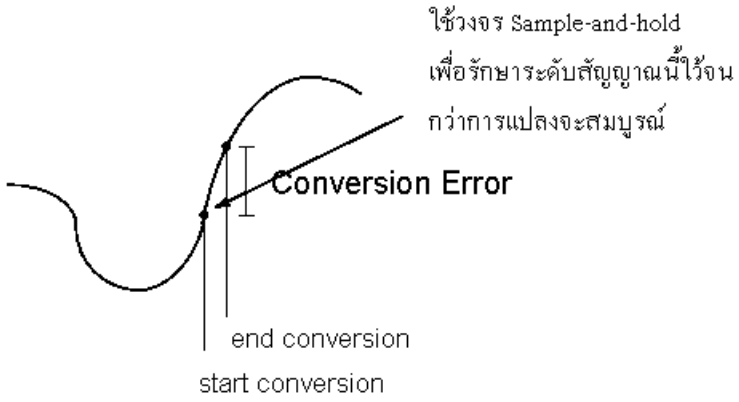
เหมือนกับใน DAC เป็นการแสดงค่าที่ถูกต้องเมื่อแทียบกันระหว่างค่าที่แท้จริงกับค่าที่กำหนดได้

Conversion Time

ระยะเวลาที่ใช้ในการแปลงสัญญาณ

Sample and Hold

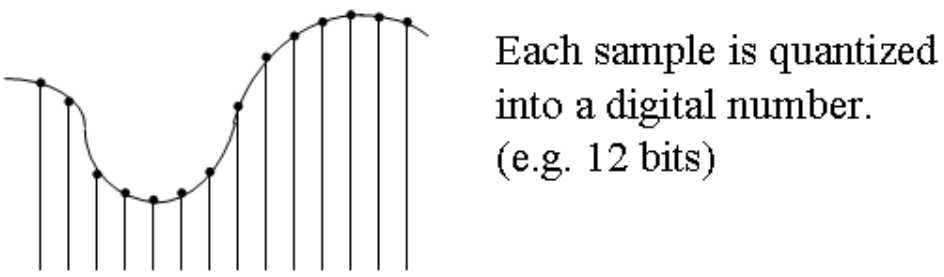
เมื่อสัญญาณมีความถี่สูงถ้าไม่ต้องการให้เกิดความผิดพลาดเนื่องจากระยะเวลาที่ใช้แปลง ควรใช้ Sample and Hold



รูปที่ 14.22 การใช้ Sample and Hold

Sampling Rate

ความถี่ของการแปลงสัญญาณ เช่นถ้าเป็น 44 kHz ก็หมายถึงว่าการแปลงสัญญาณจะเกิดขึ้นทุกๆ 22.7 μS

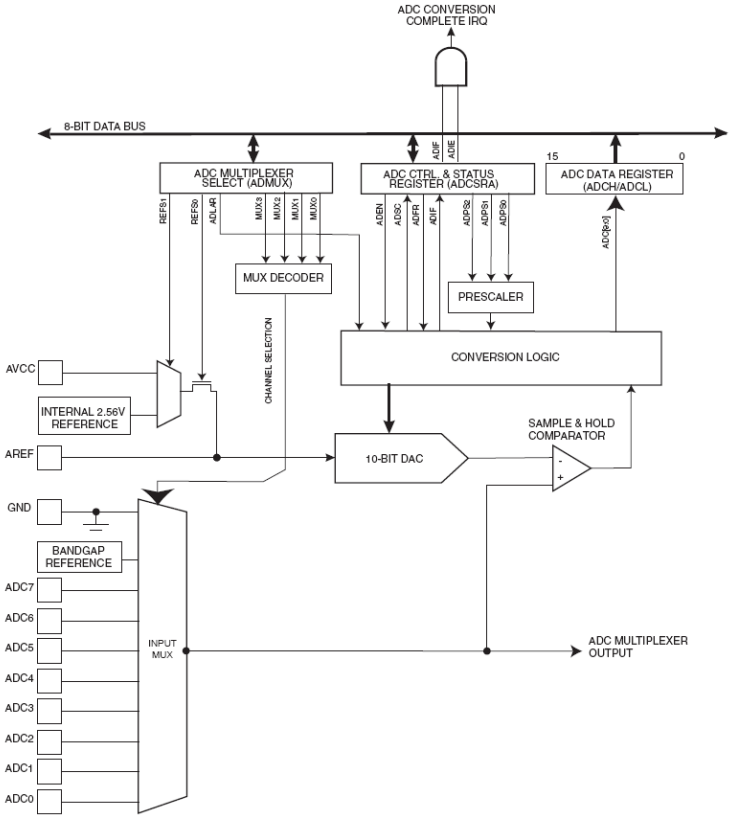


รูปที่ 14.23 ลักษณะของการสุ่มสัญญาณ

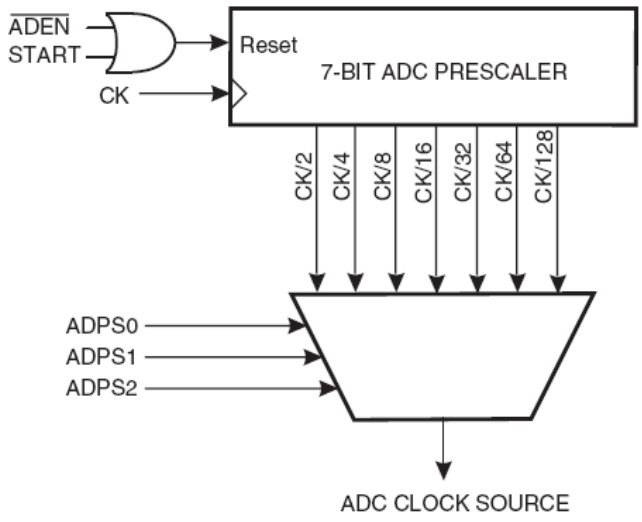
Analog-to-Digital Converter (ADC) ของ ATmega8 ขนาด 10 บิต

- 10-bit Resolution
- 0.5 LSB Integral Non-linearity
- ± 2 LSB Absolute Accuracy
- 13 - 260 µs Conversion Time
- Up to 15 kSPS at Maximum Resolution
- 6 Multiplexed Single Ended Input Channels
- 2 Additional Multiplexed Single Ended Input Channels (TQFP and QFN/MLF Package only)
- Optional Left Adjustment for ADC Result Readout
- 0 - VCC ADC Input Voltage Range
- Selectable 2.56V ADC Reference Voltage
- Free Running or Single Conversion Mode
- Interrupt on ADC Conversion Complete
- Sleep Mode Noise Canceler

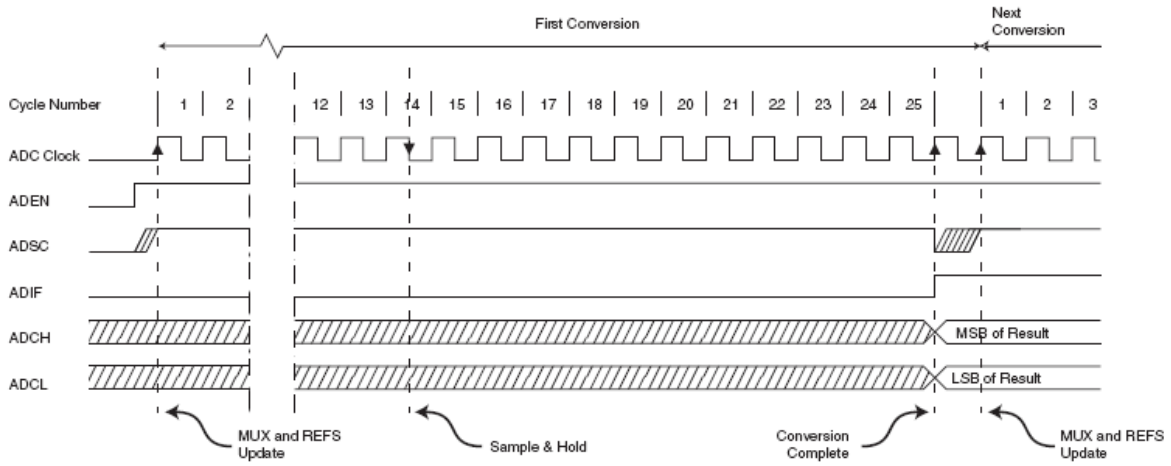
Analog to Digital Converter Block Schematic Operation



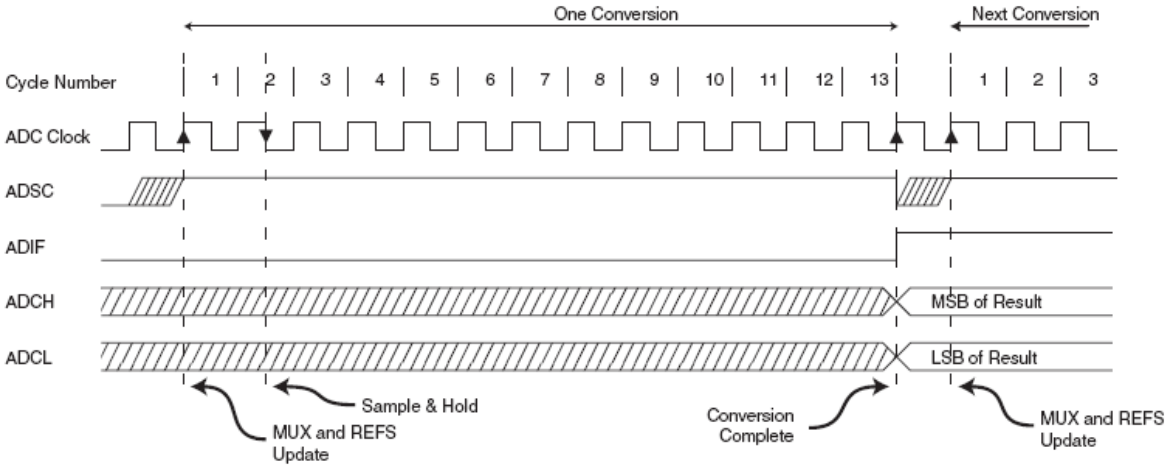
Prescaling and Conversion Timing



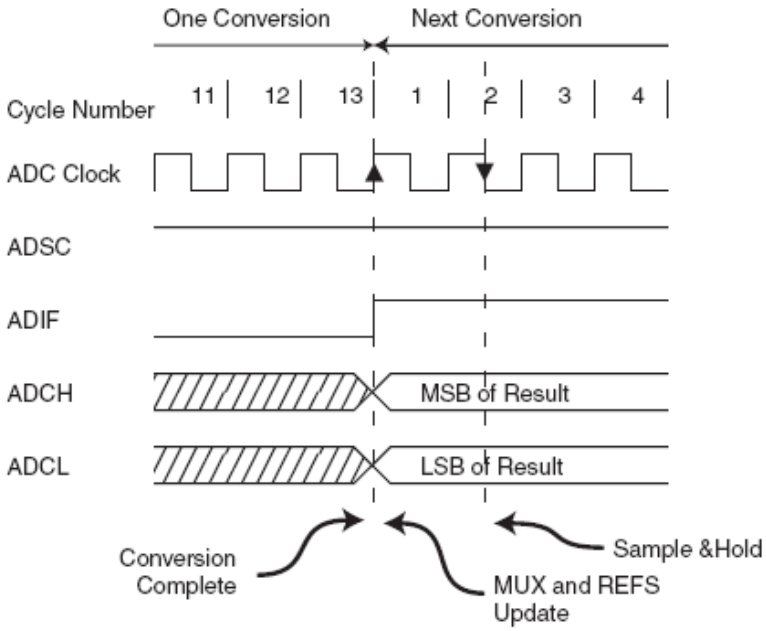
ADC Timing Diagram, First Conversion (Single Conversion Mode)



ADC Timing Diagram, Single Conversion



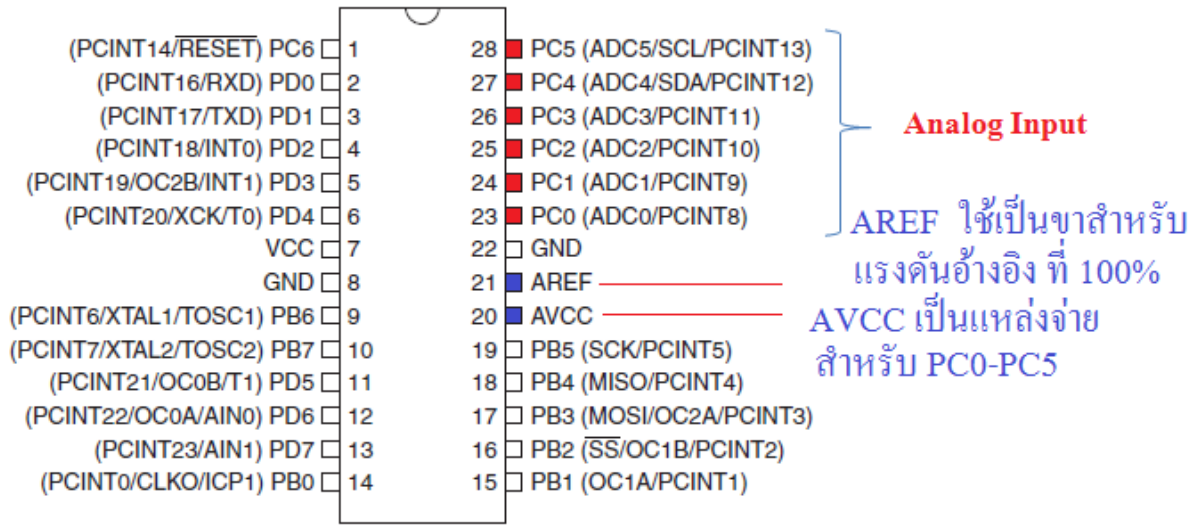
ADC Timing Diagram, Free Running Conversion



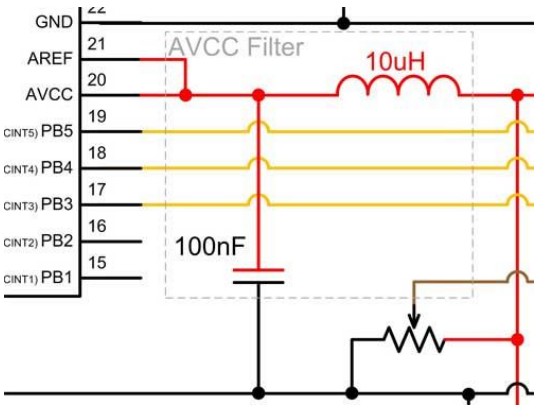
ADC Conversion Time

Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
Extended conversion	13.5	25
Normal conversions, single ended	1.5	13

ADC ใน ATMega168



แหล่งจ่ายแรงดันและแรงดันอ้างอิง



AVCC

Atmega168 มีขาแหล่งจ่ายแรงดันอยู่ 2 ขา คือ VCC และ AVCC.

AVCC เป็นแหล่งจ่ายสำหรับ PC0-PC5 เมื่อทำเป็นขาอินพุทสำหรับสัญญาณ analog แหล่งจ่าย AVCC นี้ต้องการความเสถียรมาก ดังนั้นจะใช้ low pass filter ซึ่งประกอบด้วย inductor และ capacitor.

AREF

ขา AREF ใช้เป็นขาสำหรับแรงดันอ้างอิง ที่ 100% (เมื่อสัญญาณ analog เท่ากับค่านี้ จะได้ค่าดิจิตอลเท่ากับ 1024)

Analogue Input

Atmega168 มีขาสำหรับสัญญาณ analog อยู่ 6 ขา PC0 to PC5. –ขาเหล่านี้เป็นได้ทั้ง digital I/O และ analogue input

Registers

Atmega168 มีรีจิสเตอร์ที่ใช้งานเกี่ยวกับการแปลงสัญญาณอนาลอกอยู่ 6 ตัว

Register	Description
ADMUX	ADC Multiplexer Selection Register
ADCSRA	ADC Control and Status Register A
ADCSRB	ADC Control and Status Register B
DIDR0	Digital Input Disable Register 0
ADCL	ADC Data Register – Low
ADCH	ADC Data Register – High

ADMUX

The ADMUX register allows you to control:

- The Reference Voltage
- Left adjustment of results (used for 8 bit results)
- Selection of input channel

ADMUX – ADC Multiplexer Selection Register – ADMUX

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – REFS1:0: Reference Selection Bits**

These bits select the voltage reference for the ADC, as shown in Table 22-2. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set). The internal voltage reference options may not be used if an external reference voltage is being applied to the AREF pin.

Table 22-2. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal V_{ref} turned off
0	1	AV_{CC} with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

ADMUX : ADLAR และ MUX3:0

- Bit 5 – ADLAR: ADC Left Adjust Result

กำหนดผลลัพทธ์การแปลง

1 = left adjust

0 = right adjusted

รายละเอียดดู “ADCL and ADCH – The ADC Data Register”

- Bits 3:0 – MUX3:0: Analog Channel Selection Bits

เลือกสัญญาณอินพุทที่ต้องการแปลง

Table 22-3. Input Channel Selections

MUX3:0	Single Ended Input
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	
1001	
1010	
1011	
1100	
1101	
1110	1.30V (V_{BG})
1111	0V (GND)

ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- Bit 7 – ADEN: ADC Enable

1 = enables the ADC

0 = ADC is turned off ถ้า turnoff ในขณะแปลง การแปลงจะหยุด

- Bit 6 – ADSC: ADC Start Conversion

แบบ Single Conversion mode

ให้บิตนี้เป็น 1 = สั่งให้เริ่มแปลงแบบ Free Running mode

ให้บิตนี้เป็น 1 = start the first conversion. The first conversion จะเริ่ม หลังจาก ADC enabled

เมื่อแปลงเสร็จบิตนี้จะเป็น 0

- Bit 5 – ADATE: ADC Free Running Select

1 = ทำงานแบบ Free Running mode ในโหมดนี้ จะแปลงต่อเนื่องไปตลอด

0 = หยุดการทำงานแบบ Free Running mode.

- Bit 4 – ADIF: ADC Interrupt Flag

ถูกทำให้เป็น 1 เมื่อแปลงเสร็จและข้อมูลถูกส่งไปที่ ADCL/ADCH
ใช้เป็นบิตบอกให้เกิดการอินเตอร์รัพท์ (บิต I ใน SREG ต้องเป็น 1 ด้วย)
บิตนี้จะถูก clear เมื่อการอินเตอร์รัพท์ได้รับการตอบสนอง

- Bit 3 – ADIE: ADC Interrupt Enable

1 = Enable

0 = Disable

When this bit is written to one and the I-bit in SREG is set, the ADC Conversion Complete Interruptis activated.

- Bits 2:0 – ADPS2:0: ADC Prescaler Select Bits

เลือกตัวหาร

Table 22-4. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

การกำหนดค่า ADCSRA และ ADMUX

ADCSRA

ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
1	0	0	0	0	0	0	1
Enable	Start Conversion เป็น 1 สั่ง	หยุดการทำงานแบบ Free Running mode	Interrupt Flag เป็น 1 เมื่อแปลงเสร็จ	Interrupt Enable	หาร 2		

ADMUX

REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0
0	0	0	0	0	0	0	0
AREF, internal Vref turned off	ปรับชิดขวา		ADC Channel 0				

ADCL and ADCH – The ADC Data Register

ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ตัวอย่างโปรแกรมภาษา C เขียนแบบ AVR

```
#include <avr\io.h>              // Most basic include files
#include <avr\interrupt.h>       // Add the necessary ones
#include <avr\signal.h>          // here
#include <util/delay.h>
unsigned int adc0;
int main(void)
{
   ADCSRA = (1<<ADEN)|(0<<ADATE);              // ADC Enable & Auto Trigger Disable
   ADCSRA |= (0<<ADPS2)|(0<<ADPS1)|(1<<ADPS0); // XTAL/8
   while(1)
   {
      ADMUX = 0b0000000;   //Aref,left adjust, select ADC0  (bit 2=0 bit 1 = 0 bit 0 = 0)
      ADCSRA |= (1<<ADSC); // ADC Start Conversion
      while (!(ADCSRA &(1<<ADIF)));    // Wait Coversion completes
      adc0 = ADCW;                     // Read ADC
      _delay_ms(10);
   }
}
```

ฟังก์ชั่นเกี่ยวกับสัญญาณอนาล็อก ของ Arduino

- analogReference(type)
- analogRead()
- analogWrite() – PWM

analogReference(type)

Configures the reference voltage used for analog input (i.e. the value used as the top of the input range). The options are:

- DEFAULT: the default analog reference of 5 volts (on 5V Arduino boards) or 3.3 volts (on 3.3V Arduino boards)
- INTERNAL: an built-in reference, equal to 1.1 volts on the ATmega168 or ATmega328 and 2.56 volts on theATmega8 (not available on the Arduino Mega)
- INTERNAL1V1: a built-in 1.1V reference (Arduino Mega only)
- INTERNAL2V56: a built-in 2.56V reference (Arduino Mega only)
- EXTERNAL: the voltage applied to the AREF pin (0 to 5V only) is used as the reference.

analogRead()

Description

Reads the value from the specified analog pin. The Arduino board contains a 6 channel (8 channels on the Mini and Nano, 16 on the Mega), 10-bit analog to digital converter. This means that it will map input voltages between 0 and 5 volts into integer values between 0 and 1023. This yields a resolution between readings of: 5 volts / 1024 units or, .0049 volts (4.9 mV) per unit. The input range and resolution can be changed using analogReference().

It takes about 100 microseconds (0.0001 s) to read an analog input, so the maximum reading rate is about 10,000 times a second.

Syntax

analogRead(pin)

เช่น int sensorValue = analogRead(A0);

และการแปลงค่าดิจิตอลที่อ่านได้ หาได้จาก float voltage= sensorValue * (5.0 / 1023.0);

ตัวอย่างโปรแกรมอ่านค่าอนาล็อกของ Arduino แล้วส่งค่าที่อ่านได้ออกทางพอร์ทอนุกรม

```
int analogPin = 3;         // potentiometer wiper (middle terminal) connected to analog pin 3
                           // outside leads to ground and +5V
int val = 0;               // variable to store the value read
void setup()
{
        Serial.begin(9600); // setup serial
}
void loop()
{
        val = analogRead(analogPin); // read the input pin
        Serial.println(val); // debug value
}
```

analogWrite() - PWM

analogWrite(pin, value)

Parameters

- pin: the pin to write to.

  กรณี atmega168 มี D3 D5 D6 D9 D10 และ D11

- value: the duty cycle: between 0 (always off) and 255 (always on).

ตัวอย่างสร้าง PWM ด้วย analogWrite

```
#define pwm_1 3
```

```
#define pwm_2 5
#define pwm_3 6
#define pwm_4 9
#define pwm_5 10
#define pwm_6 11

void setup()
{
  analogWrite(pwm_1, 10);
  analogWrite(pwm_2, 50);
  analogWrite(pwm_3, 100);
  analogWrite(pwm_4, 150);
  analogWrite(pwm_5, 200);
  analogWrite(pwm_6, 250);
 }

void loop()
{

}
```