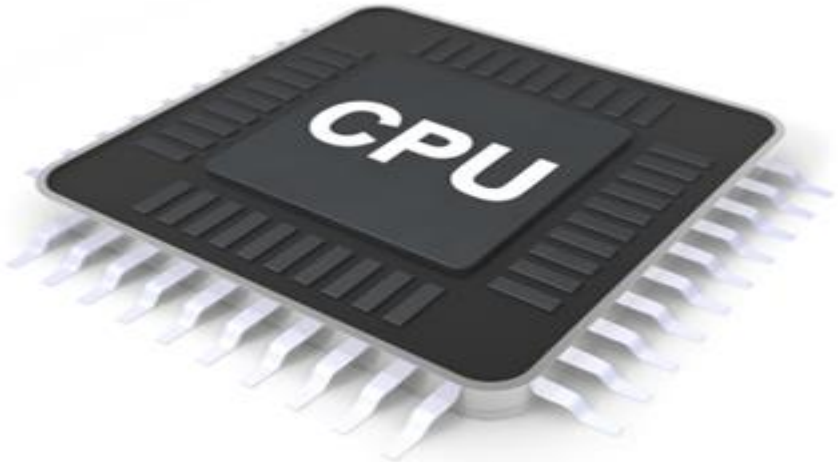


Digital to Analog and Analog to Digital

ENEE 3587

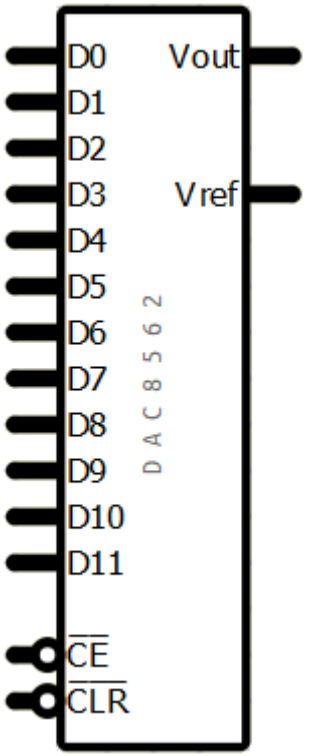
Microp

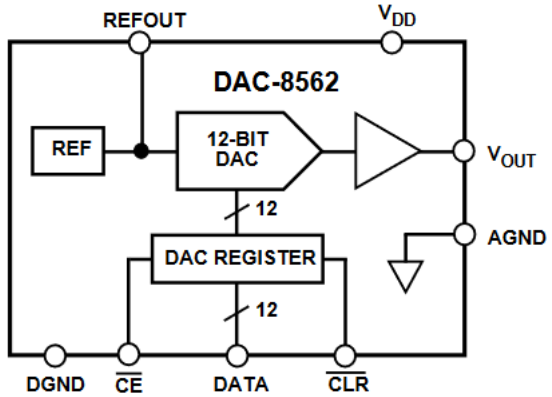
Digital vs Analog

- Many systems in the real work are simple analog system
- MCU doesn’t have a Digital to analog conversion (DAC) module.
- Analog vs Digital signals
  - Analog: continuous time, discontinuous/continuous ampltitude
  - Digital: discrete time, discrete amplitude
- MCU are digital
  - Clock driven systems.
  - Accuracy and speed.
  - Noise protection
- Many real word systems are analog
  - Many electric/mechanical systems
  - Sensory based systems: temperature, pressure, vision, audio

12-bit DAC: DAC8562

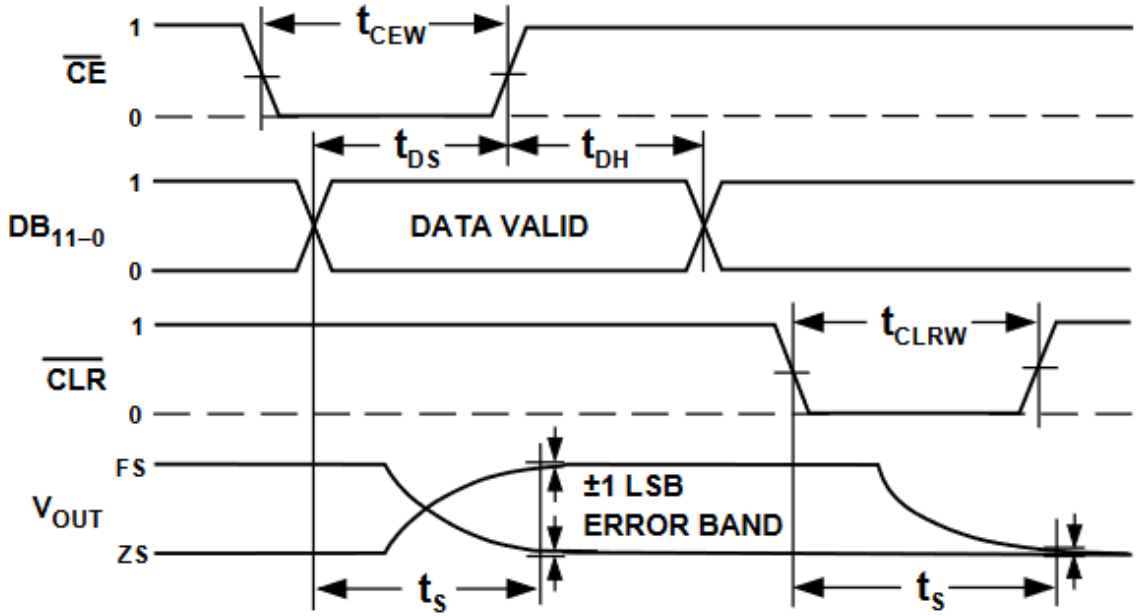
- 12-bit parallel-input Digital to Analog Converter (DAC):
  - https://www.analog.com/media/en/technical-documentation/data-sheets/DAC8562.pdf
- Pins:
  - **DB0-DB11**: Data inputs, DB11 is MSB
  - **CE**: Chip Enable. Active low input.
  - **CLR**: Active low. Clears DAC register to zero
  - **DGND**: Digital ground for input logic.
  - **AGND**: Analog Ground reference
  - **VOUT**: Voltage output
  - **REFOUT**: Nominal (2.5 V) reference





DAC8562 Specifications

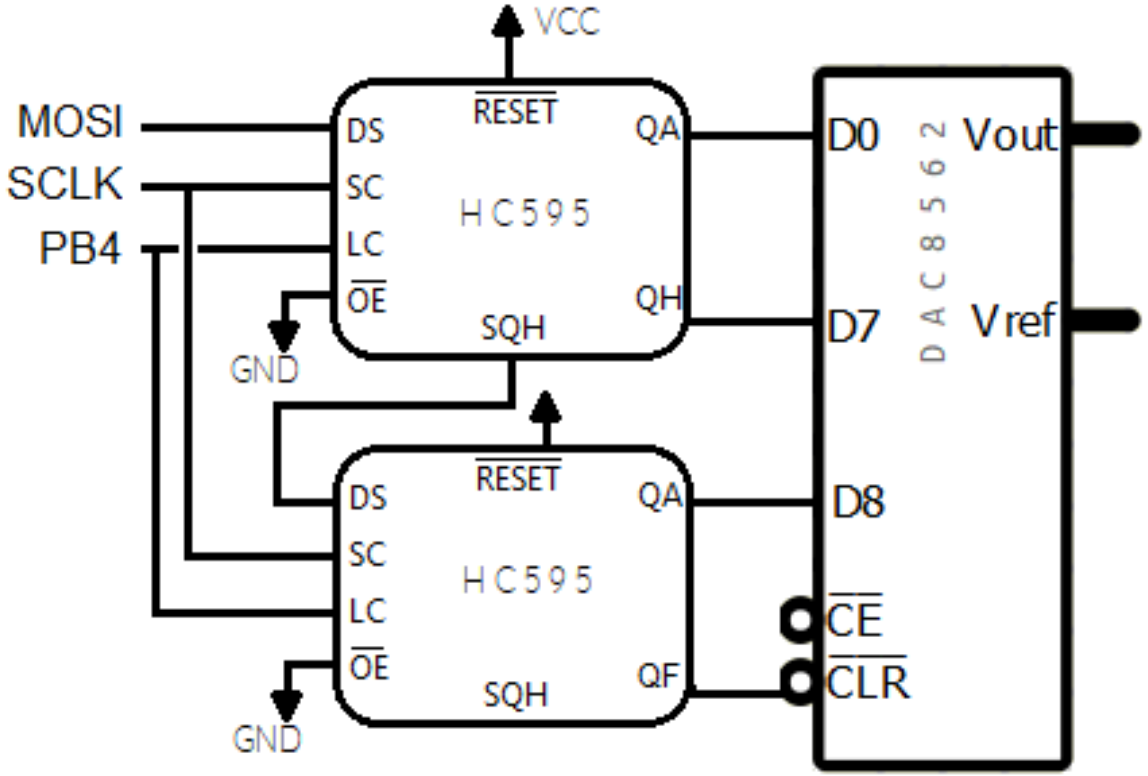
- $V_{out} = \left(\frac{DATA}{2^{12}-1}\right) V_{ref} = \left(\frac{DATA}{4095}\right) V_{ref}$
  - DATA = 0x0000 = 0 => Vout = 0
  - DATA = 0x0FFF = 4095 => Vout = Vref
- $Data = \frac{V_{out}}{V_{ref}} 4095$
- Min SPI Clock: 2/16MHz = 125ns



		Min	Typ
Chip Enable Pulse Width	t_{CEW}	30 ns	
Data Setup	t_{DS}	30 ns	
Data Hold	t_{DH}	10 ns	
Clear Pulse Width	t_{CLRW}	20 ns	
Voltage Settling Time	t_S		16 µs

Interfacing DAC with SPI

- Use 2x shift registers to send data to
  - Sample time = 16/SPI CLK
  - Nyquist: signal max freq < sampling freq/2
  - $f_{signal} < f_{SPI}/32$
- Shift register 1: QA-QH: D0-D7
- Shift register 2:
  - QA-QD: D8-D11
  - QE: CE
  - QF: CLR
  - QH: not needed



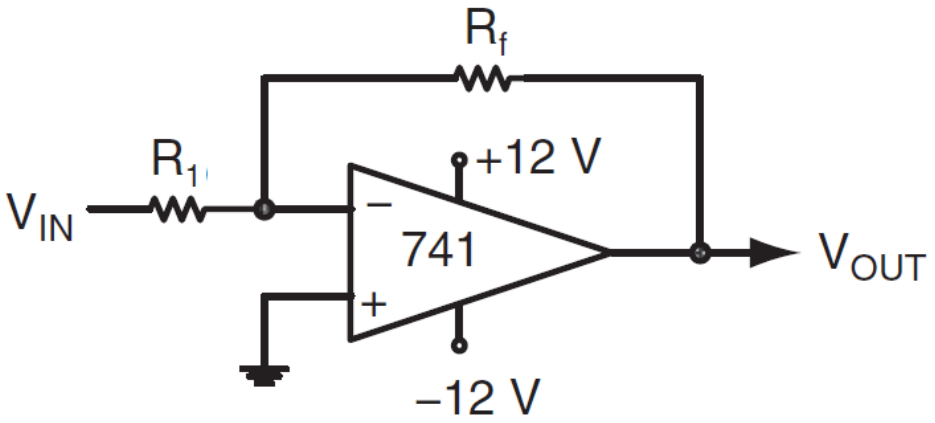
ADC

- Most MCU use successive-approximation ADC
  - Analog difference circuit
  - Needs a HI/LO voltage references: V_{RH}, V_{RL}
  - n-bit output: represents V_{RL} to V_{RH} in steps of $(V_{RH} - V_{RL})/2n$
- Input signals must be conditioned
  - Current must not exceed ADC max/min
  - Voltage must be within V_{RL}, V_{RH}
  - Use resistors, OPAMPs to condition

OPAMPs

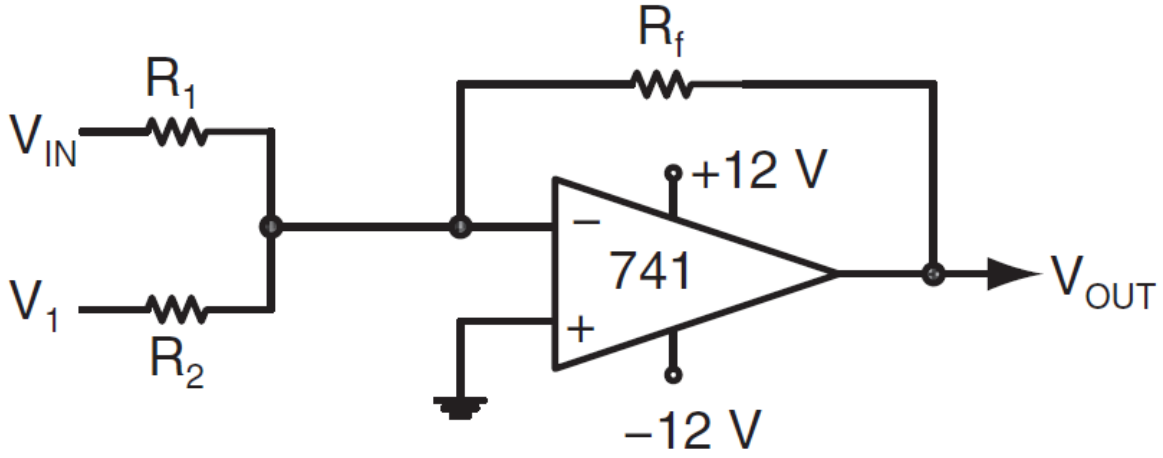
- Inverting:

$$V_{OUT} = -\frac{R_f}{R_1} Vin$$



- Summing:

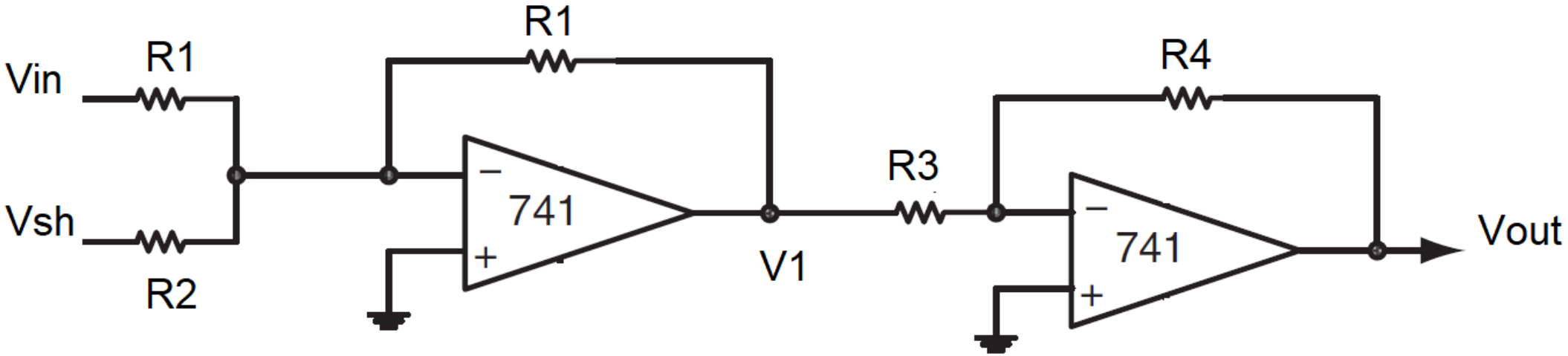
$$V_{OUT} = -\left(\frac{R_f}{R_1} Vin + \frac{R_f}{R_2} V_1\right)$$



Signal Conditioning

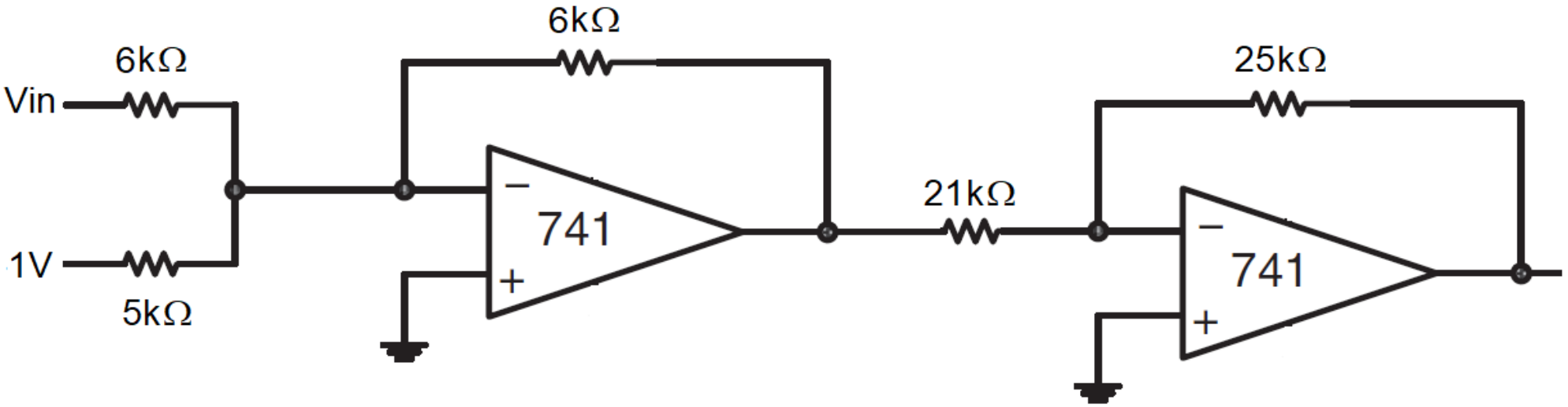
- Given analog circuit output is -1.2 V $\rightarrow$ 3.0 V and V_{RL}-V_{RH} is 0V – 5V. Condition the signal.
- Solution:
  - Signal range: 3 – (-1.2) = 4.2V
  - Input range is 5 – 0 = 5V
  - Scale: 5/4.2 = 1.5625
    - Convert to a fraction:
      5/(42/10) = 50/42 = 25/21
  - Shift: 0-(-1.2) = 1.2V
    - Convert to a fraction: 12/10=6/5
  - Shift up by 1.2 then scale by 1.5625

OPAMP Conditioning



- First, apply shift 1.2V:
  - Vsh = 1V
  - 1.2 = 6/5 => R1 = 6, R2 = 5
- Next, apply scale:
  - 25/21 => R3 = 21, R4 = 25

OPAMP Circuit



$$-\frac{6}{5}1\text{V}-\frac{6}{6}\text{V}_{\text{in}}=-1.2-\text{V}_{\text{in}}=\text{V}_1$$

$$\text{V}_{\text{in}}=-1.2\text{V}\rightarrow 3\text{V}$$

$$-1.2-\text{V}_{\text{in}}=-1.2-(-1.2\text{V}\rightarrow 3\text{V})$$

$$=-1.2-3V\rightarrow 1.2$$

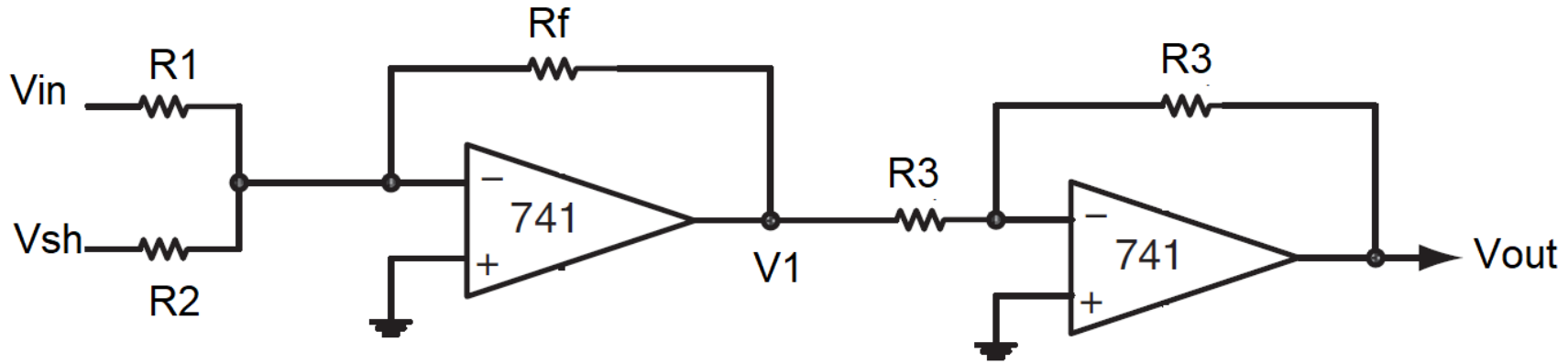
$$\text{V}_1=-4.2\rightarrow 0$$

$$-\frac{25}{21}\text{V}_1=\text{Vout}$$

$$-\frac{25}{21}(-4.2\rightarrow 0)=5\text{V}\ \rightarrow 0$$

Alternative OPAMP Conditioning

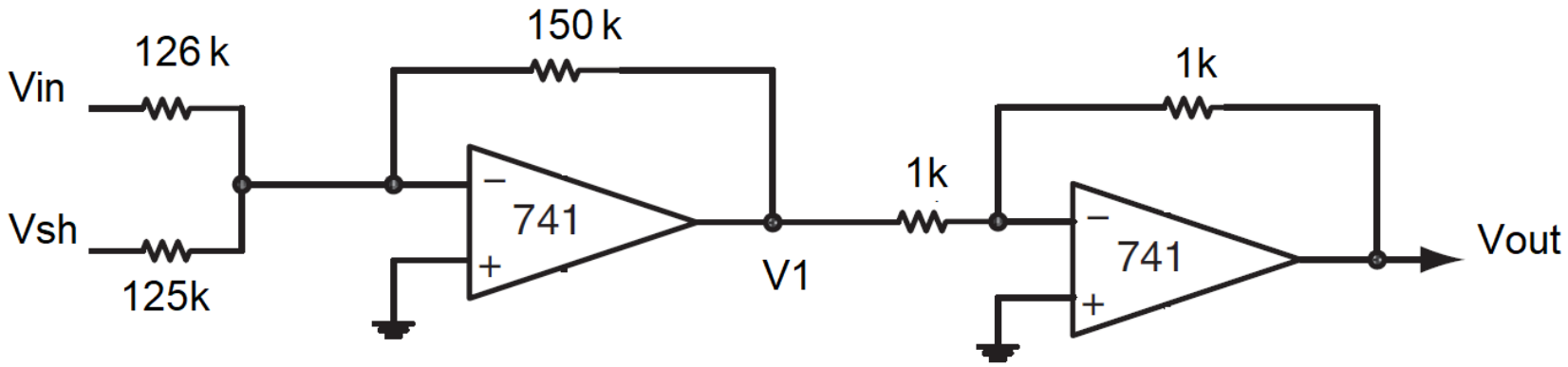
- Combine scaling and shifting together, followed by inversion.



$$V_{\text{out}} = -\frac{\text{R3}}{\text{R3}}\left(-\frac{\text{Rf}}{\text{R2}}\text{V}_{\text{sh}} - \frac{\text{Rf}}{\text{R1}}\text{V}_{\text{in}}\right) = -\left(-\frac{\text{Rf}}{\text{R2}}\text{V}_{\text{sh}} - \frac{\text{Rf}}{\text{R2}}\text{V}_{\text{in}}\right) = \frac{\text{Rf}}{\text{R2}}\text{V}_{\text{sh}} + \frac{\text{Rf}}{\text{R1}}\text{V}_{\text{in}}$$

Alternative OPAMP Circuit

- For -6/5V shift and 25/21 scale:
  - Rf = 6*25 = 150
  - R1 = 5*25 = 125
  - R2 = 21*6 = 126, Vsh = -1V
  - R3 = 1



$$V_{out} = \frac{Rf}{R2}V_{sh} + \frac{Rf}{R1}V_{in} = \frac{150}{125}(-1V) + \frac{150}{126}V_{in} = -1.2 + \frac{25}{21}V_{in}$$

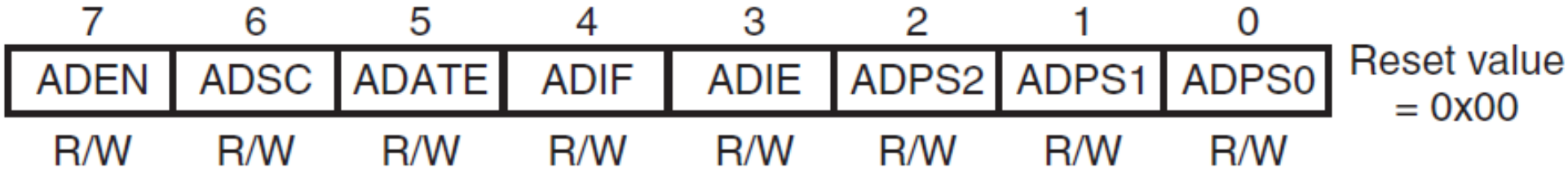
ATmega ADC

- 10-bit ADC, 1024 levels
- 13µs - 260µs Conversion Time
  - 76.9kSPS to 15kSPS
- successive approximation method
- Single-ended: 8 or 16 input channels
  - ADC pins 7:0 connected to Port F pins 7:0,
  - ADC pins 16:8 connected to Port K pins 7:0
  - 0V – VCC
- Differential A/D: 14 differential input channels
  - 2.7V – VCC
  - 4 channels with optional Gain of 1x, 10x, 200x
- Vcc selectable: 2.56V or 1.1V

ADC Registers

- ADCH:ADCL — ADC data register high and low
- ADCSRA, ADCSRB — ADC Control and Status Register
- ADMUX — ADC multiplexer selection register
- SPIOR — Special Function I/O Register
- DIDR0, DIDR1 — Digital Input Disable Registers 0 and 1

ADCSRA



ADEN: ADC enable

1: ADC module enabled

ADSC: ADC start conversion

1: starts conversion.

ADATE: ADC auto trigger enable

1: conversion on a positive edge of the selected trigger signal.

ADIF: ADC interrupt flag

1: ADC conversion completed. Cleared by writing a 1 to it.

ADIE: ADC interrupt enable

1 = ADC interrupt enabled

ADPS2-ADPS0: ADC prescaler select bits

Divisor of ADC clock.

ADPS2	ADPS1	ADPS0	Prescaler
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

ADCSRB

7	6	5	4	3	2	1	0	
—	ACME	—	—	MUX5	ADTS2	ADTS1	ADTS0	Reset value = 0x00
R	R/W	R	R	R/W	R/W	R/W	R/W	

ACME: Analog Comparator Multiplexer Enable

0: ADC uses AIN for negative

1: ADC uses negative input

MUX5: Analog channel select bit 5

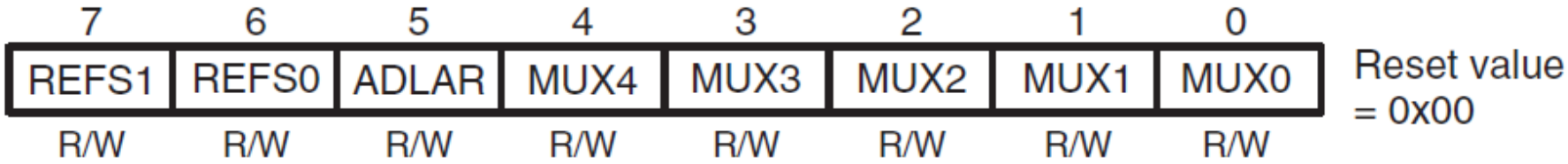
Combined with MUX4-MUX0 from ADMUX

ADTS2-ADTS0: ADC auto trigger source

ADATE bit in the **ADCSRA** register is set

ADTS2	ADTS1	ADTS0	Trigger source
0	0	0	Free running mode
0	0	1	Analog comparator
0	1	0	External interrupt request 0
0	1	1	Timer/counter0 compare match A
1	0	0	Timer/counter0 overflow
1	0	1	Timer/counter1 compare match B
1	1	0	Timer/counter1 overflow
1	1	1	Timer/counter1 capture event

ADMUX



REFS1:0: Reference selection bits

These two bits select the voltage reference

REFS1	REFS0	Voltage reference selection
0	0	AREF, internal VREF turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Internal 1.1 V voltage reference
1	1	Internal 2.56 V voltage reference

ADLAR: ADC left adjust result

0 = ADC result right justified

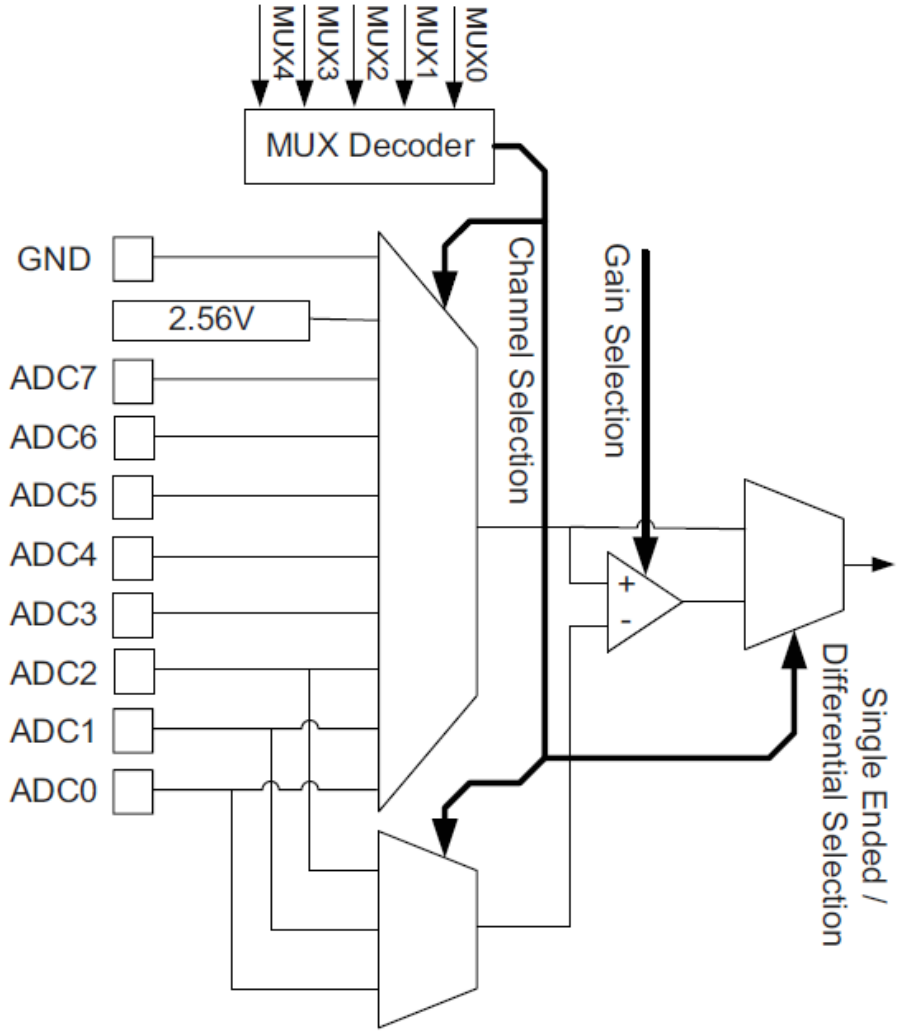
1 = ADC result left justified

Changing this bit will affect the ADC result register immediately.

MUX4-MUX0: Analog channel and gain selection bits

The value of these bits select the analog input combination to connect to ADC.

Single-Ended vs Differential Input



MUX5:0 Single-Ended Input

- Data register connected to 1 channel

MUX5:0	Data Register Connected to Channel
000000	ADC0
000001	ADC1
000010	ADC2
000011	ADC3
000100	ADC4
000101	ADC5
000110	ADC6
000111	ADC7
011110	1.1V
011111	0V

MUX5:0	Data Register Connected to Channel
100000	ADC8
100001	ADC9
100010	ADC10
100011	ADC11
100100	ADC12
100101	ADC13
100110	ADC14
100111	ADC15

MUX5:0 Differential Input

- Data register contains the difference between 2 a positive and a negative
  - Positive can be any channel
  - For ADC7:0, negative can be ADC0, ADC1, ADC2
  - For ADC15:8, negative can be ADC8, ADC9, ADC10
  - Don’t choose the same channel as positive and negative
  - Gain can be 1x, 10x, 200x

MUX5:0	Positive differential input	Negative differential input	Gain
001000(1)	ADC0	ADC0	10
001001(1)	ADC1	ADC0	10
001010(1)	ADC0	ADC0	200
001011(1)	ADC1	ADC0	200
001100(1)	ADC2	ADC2	10
001101(1)	ADC3	ADC2	10
001110(1)	ADC2	ADC2	200
001111(1)	ADC3	ADC2	200
010000	ADC0	ADC1	1
010001	ADC1	ADC1	1
010010	ADC2	ADC1	1
010011	ADC3	ADC1	1
010100	ADC4	ADC1	1
010101	ADC5	ADC1	1
010110	ADC6	ADC1	1
010111	ADC7	ADC1	1
011000	ADC0	ADC2	1
011001	ADC1	ADC2	1
011010	ADC2	ADC2	1
011011	ADC3	ADC2	1
011100	ADC4	ADC2	1
011101	ADC5	ADC2	1

MUX5:0	Positive differential input	Negative differential input	Gain
101000(1)	ADC8	ADC8	10
101001(1)	ADC9	ADC8	10
101010(1)	ADC8	ADC8	200
101011(1)	ADC9	ADC8	200
101100(1)	ADC10	ADC10	10
101101(1)	ADC11	ADC10	10
101110(1)	ADC10	ADC10	200
101111(1)	ADC11	ADC10	200
110000	ADC8	ADC9	1
110001	ADC9	ADC9	1
110010	ADC10	ADC9	1
110011	ADC11	ADC9	1
110100	ADC12	ADC9	1
110101	ADC13	ADC9	1
110110	ADC14	ADC9	1
110111	ADC15	ADC9	1
111000	ADC8	ADC10	1
111001	ADC9	ADC10	1
111010	ADC10	ADC10	1
111011	ADC11	ADC10	1
111100	ADC12	ADC10	1
111101	ADC13	ADC10	1

DIDR0, DIDR1

7	6	5	4	3	2	1	0	
ADC7D	ADC6D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	Reset value = 0x00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

(a) Digital input disable register 0 (DIDR0)

7	6	5	4	3	2	1	0	
ADC15D	ADC14D	ADC13D	ADC12D	ADC11D	ADC10D	ADC9D	ADC8D	Reset value = 0x00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

(b) Digital input disable register 1 (DIDR1)

- Digital input disable
  - 0 = Digital input enabled
  - 1 = Digital input disabled.
- Disable digital input buffer to reduce power consumption.

Conversion Values and Times

- Conversion value:

$$\text{ADC} = \frac{V_{in}}{V_{Ref}} 1024$$

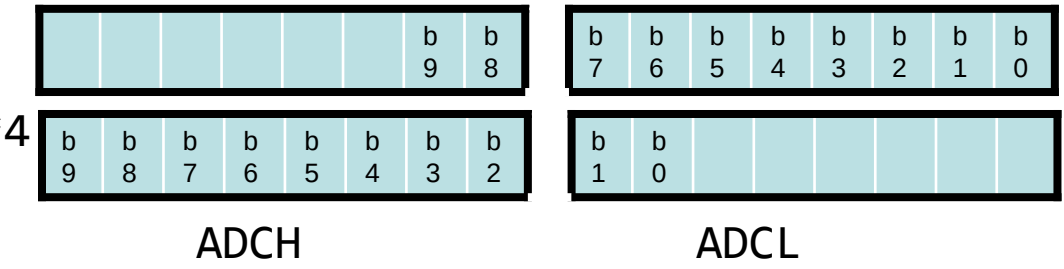
$$\text{ADC} = \frac{V_{POS} - V_{NEG}}{V_{Ref}} 512 \times \text{Gain}$$

- Conversion time

Condition	Sample and hold (cycles from start of conversion)	Conversion time (cycles)
First conversion	13.5	25
Normal conversions, single ended	1.5	13
Auto triggered conversions	2	13.5
Normal conversions, differential	1.5/2.5	13/14

ADC Setup: Polling for 1 Conversion

1. Configure:
   - PORT pin for input
   - Enable ADC module: `ADCSRA = 0b1------`
   - Select conversion speed based on application: `ADCSRA = 0b----XXX`
   - Select Vref We use the REFS1:0: `ADMUX = 0bXX-----`
2. Select input channel,
   - MUX4:0: `ADMUX = 0b---XXXXX`
   - MUX5: `ADCSRB = 0b----X---`
3. Activate conversion: `ADCSRA = 0b-1------`
4. Wait for the conversion to complete: `while(!(ADCSRA==0x10))`
5. Read the ADCL, ADCH
   - read ADCL before ADCH
   - Right justified: `value = (int)ADCL + (int)ADCH*256`
   - Left justified: `value = (int)ADCL/128 + (int)ADCH*4`



Coding Example

- Read signal connected to ADC0 pin. Assume AVCC = VCC = 5V and f_{OSC} = 16 MHz. The signal can vary from 0 V to 5 V.

```
DDRF &= 0xFE;    // ADC0 connected to Port F pin 0
ADCSRA = 0x87;   // enable ADC, disable auto-trigger, clock=125 kHz
ADMUX = 0x40;    // AVCC as reference voltage, ADC0, right justified
ADCSRB = 0;      // single-ended channel
DIDR0 = 0xFE;    // disable digital bufs for all except ADC0 input
DIDR1 = 0xFF;

ADCSRA |= 0x40;                 // start an A/D conversion
while(!(ADCSRA & 0x10));        // wait for A/D conversion to complete
ADCSRA |= 0x10;                 // clear ADIF flag
temp = (int)ADCL + (int)(ADCH*256); // combine the upper and lower
```