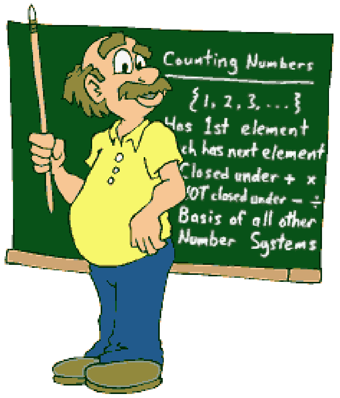
Ingeniørhøjskolen i Århus

IECA

Embedded Computer Architecture

Lesson 19: A/D Converting



Version: 16-11-2015, Henning Hargaard

Why ADC?

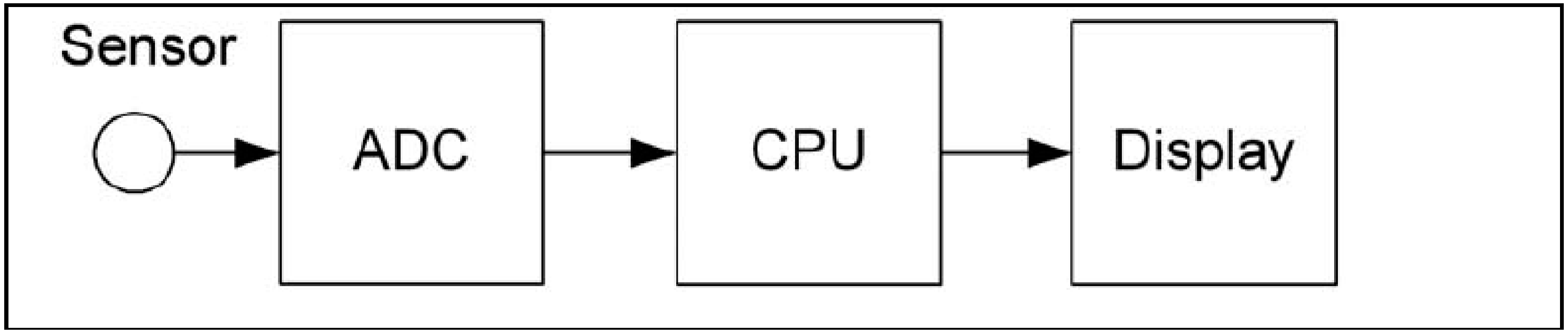
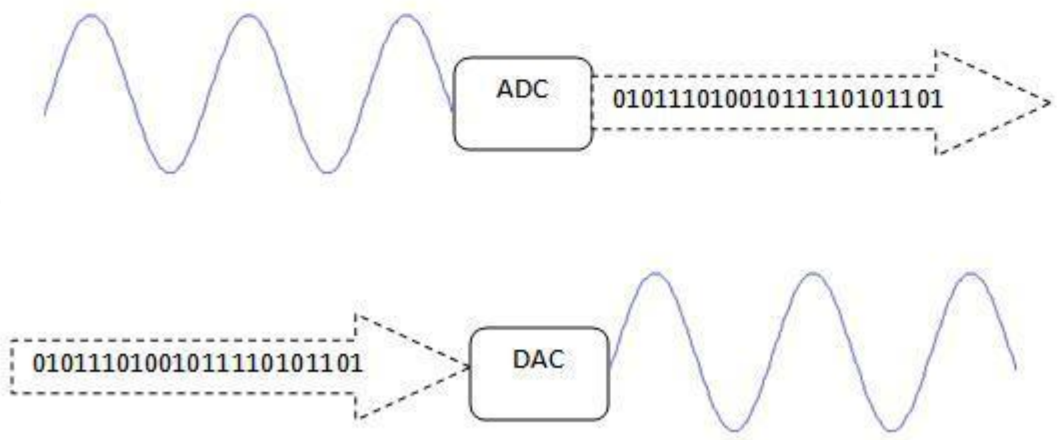
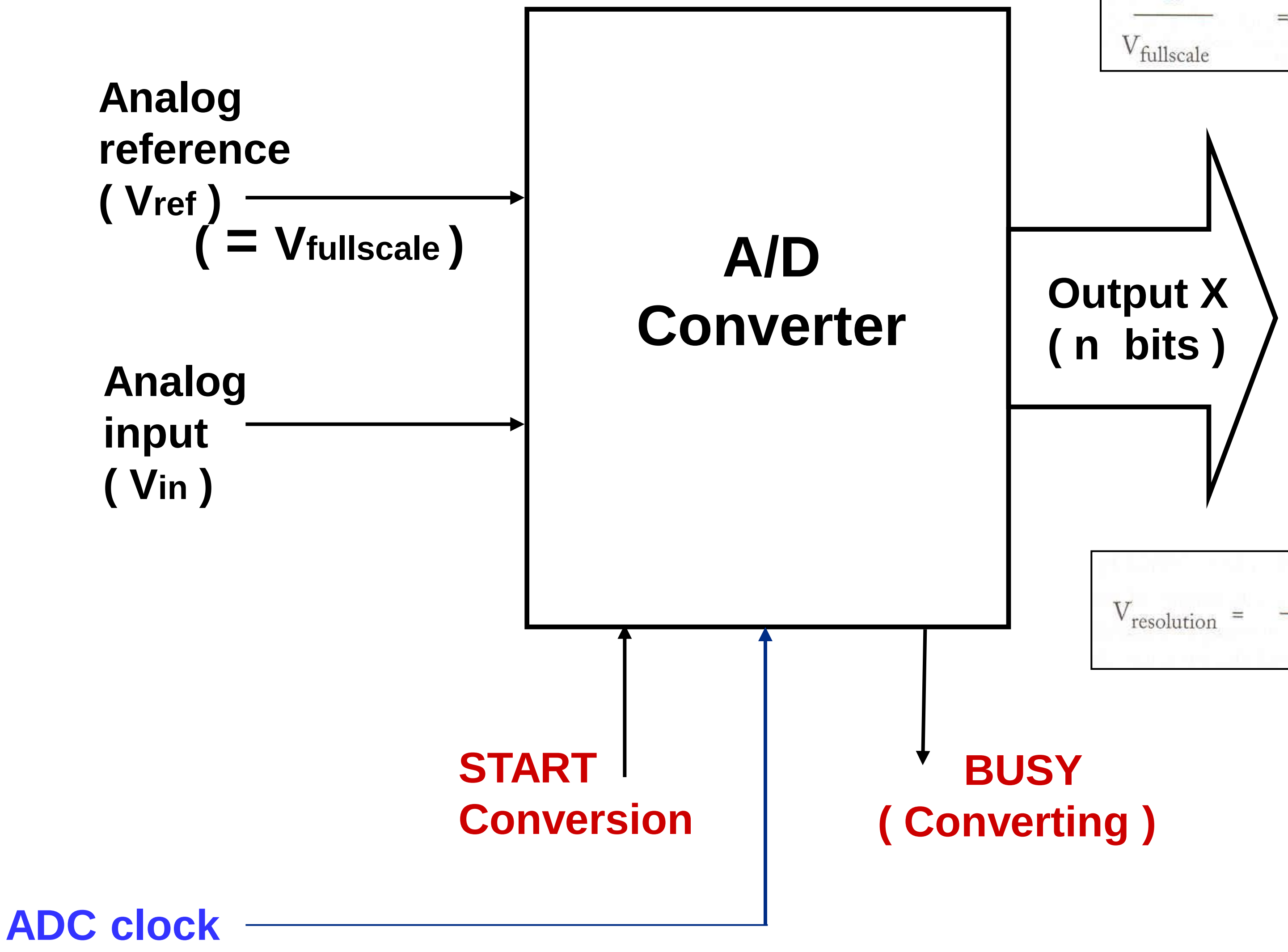


Figure 13-1. Microcontroller Connection to Sensor via ADC



ADC in general



$$\frac{V_{in}}{V_{fullscale}} = \frac{X}{2^n-1}$$

$$V_{resolution} = \frac{V_{fullscale}}{2^n-1}$$

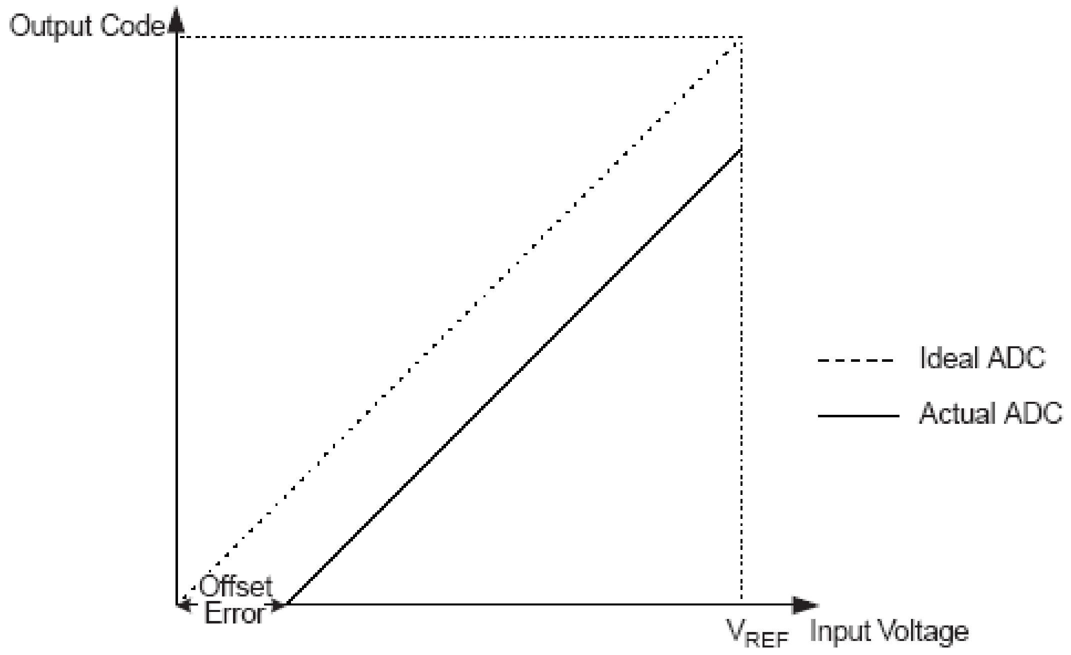
Test (”socrative.com”: Room = MSYS)

An A/D converter reference voltage is 5 volts, and the result is represented by 12 bits.

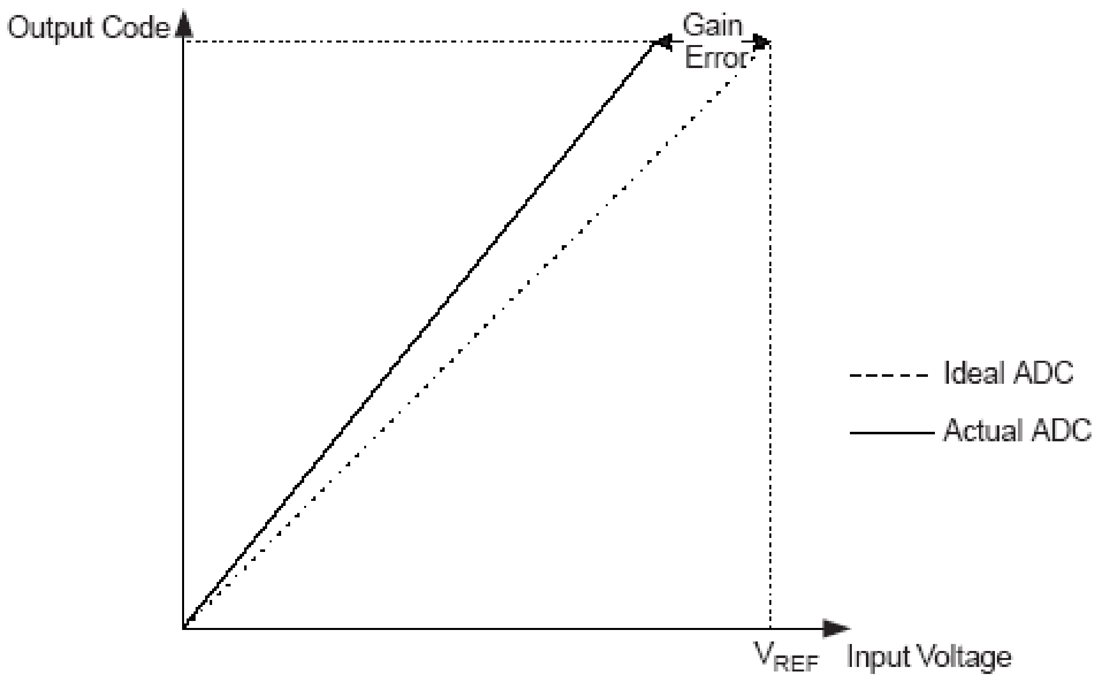
What’s the resolution for the A/D converter ?

- A: 1,22 mV
- B: 12 mV
- C: 4,88 mV
- D: 5,12 mV

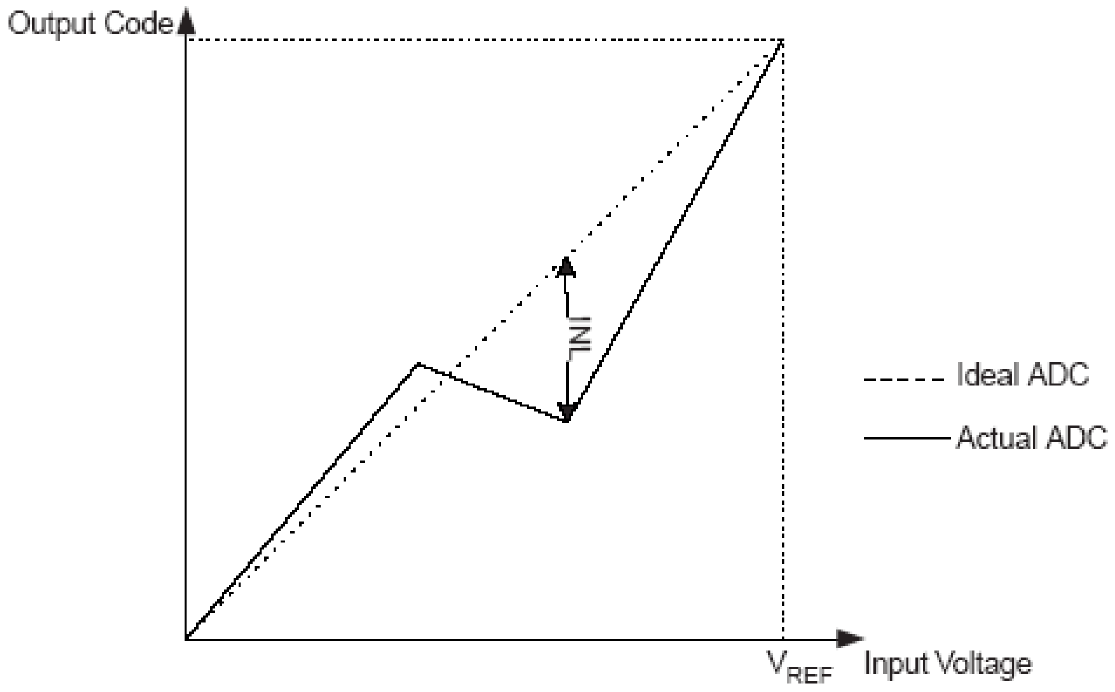
Offset error



Gain error


Output Code
Gain Error
Ideal ADC
Actual ADC
V_{REF}
Input Voltage

Integral non-linearity



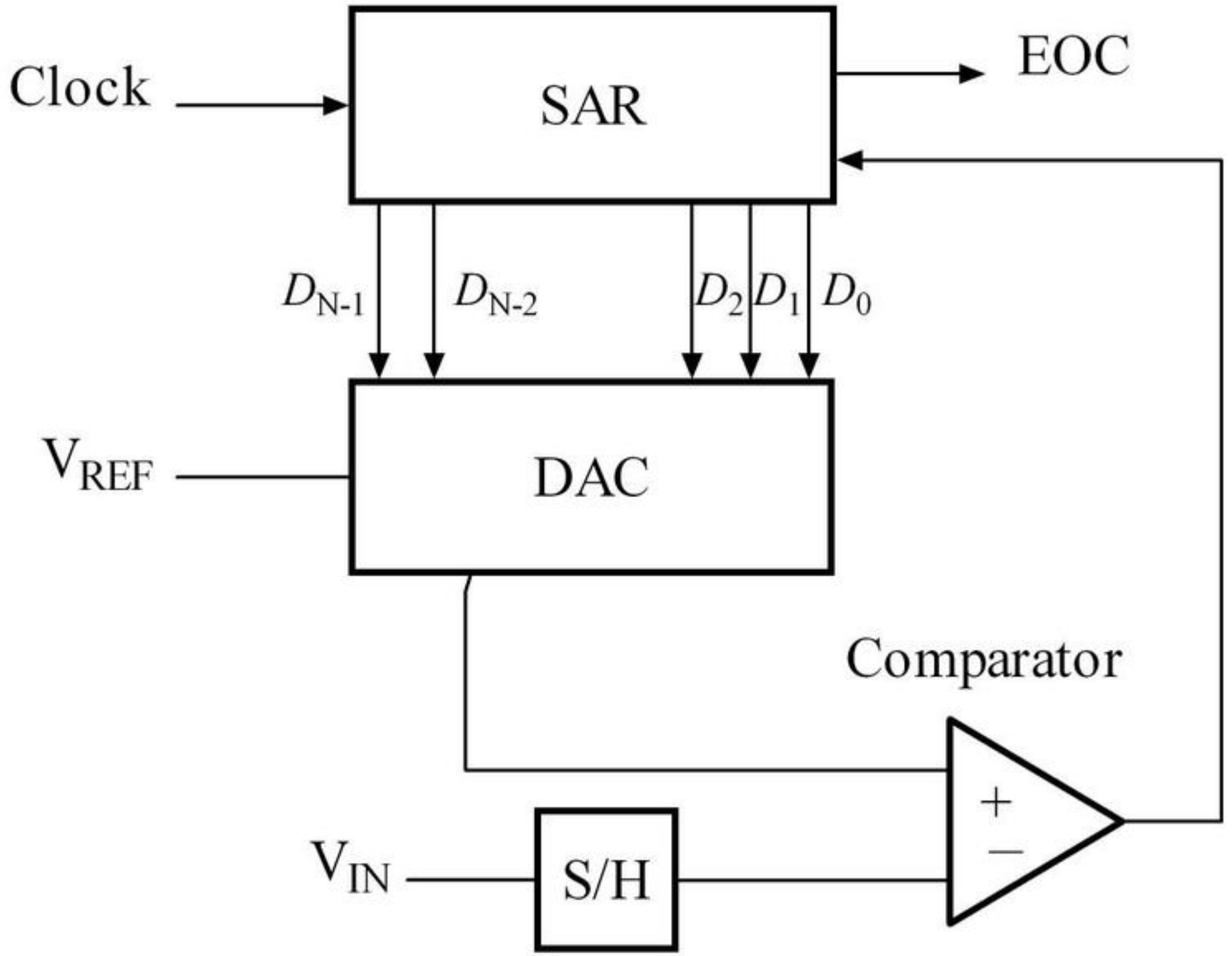
Mega32 ADC : Features

- 10-bit Resolution
- 0.5 LSB Integral Non-linearity
- ±2 LSB Absolute Accuracy
- 65 - 260 µs Conversion Time
- Up to 15 kSPS at Maximum Resolution
- 8 Multiplexed Single Ended Input Channels
- 7 Differential Input Channels
- 2 Differential Input Channels with Optional Gain of 10x and 200x(1)
- Optional Left adjustment for ADC Result Readout
- 0 - V_{CC} ADC Input Voltage Range
- Selectable 2.56V ADC Reference Voltage
- Free Running or Single Conversion Mode
- ADC Start Conversion by Auto Triggering on Interrupt Sources
- Interrupt on ADC Conversion Complete

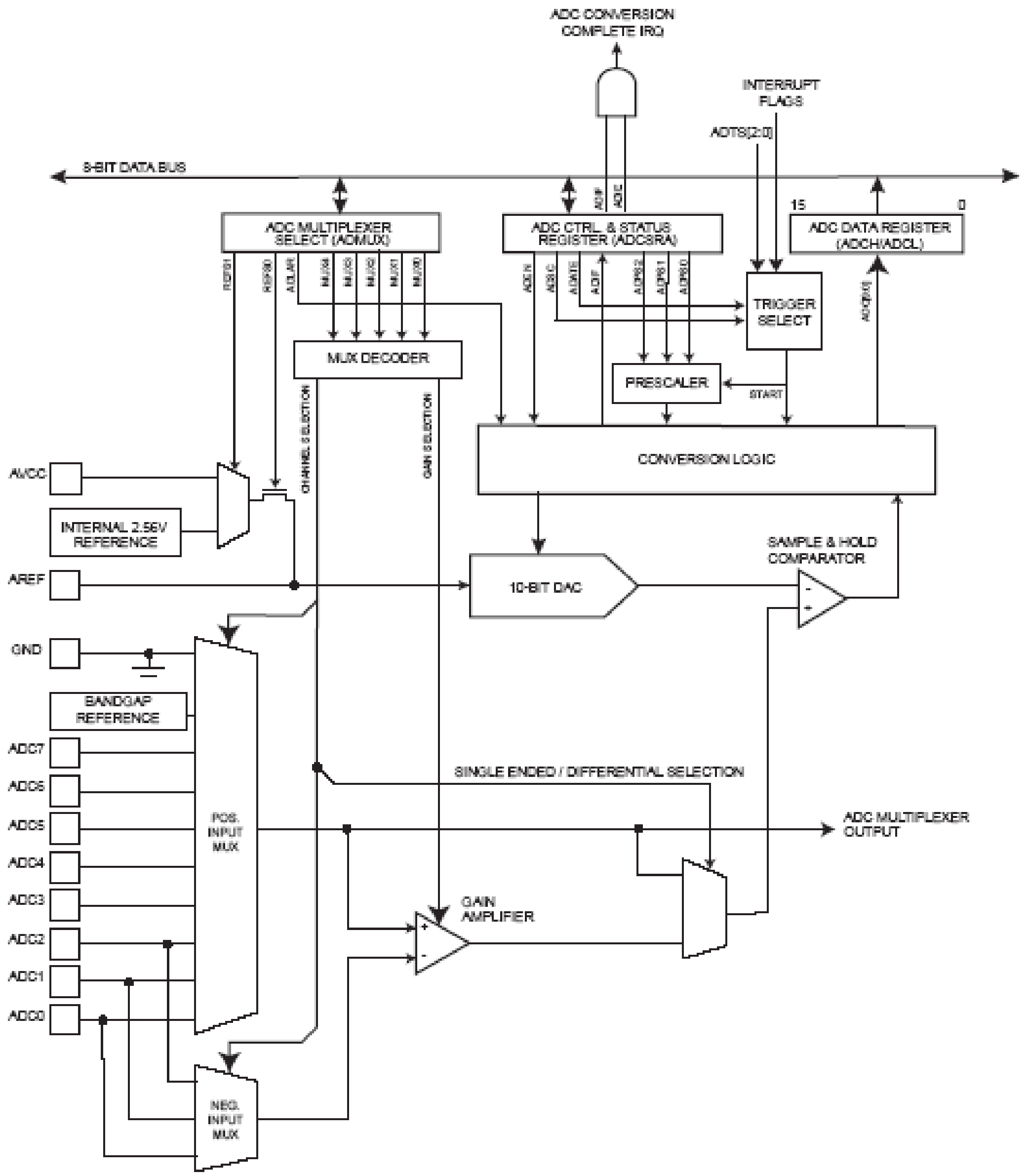
Mega32 ADC

Mega32 ADC = ”Successiv Approximation ADC”

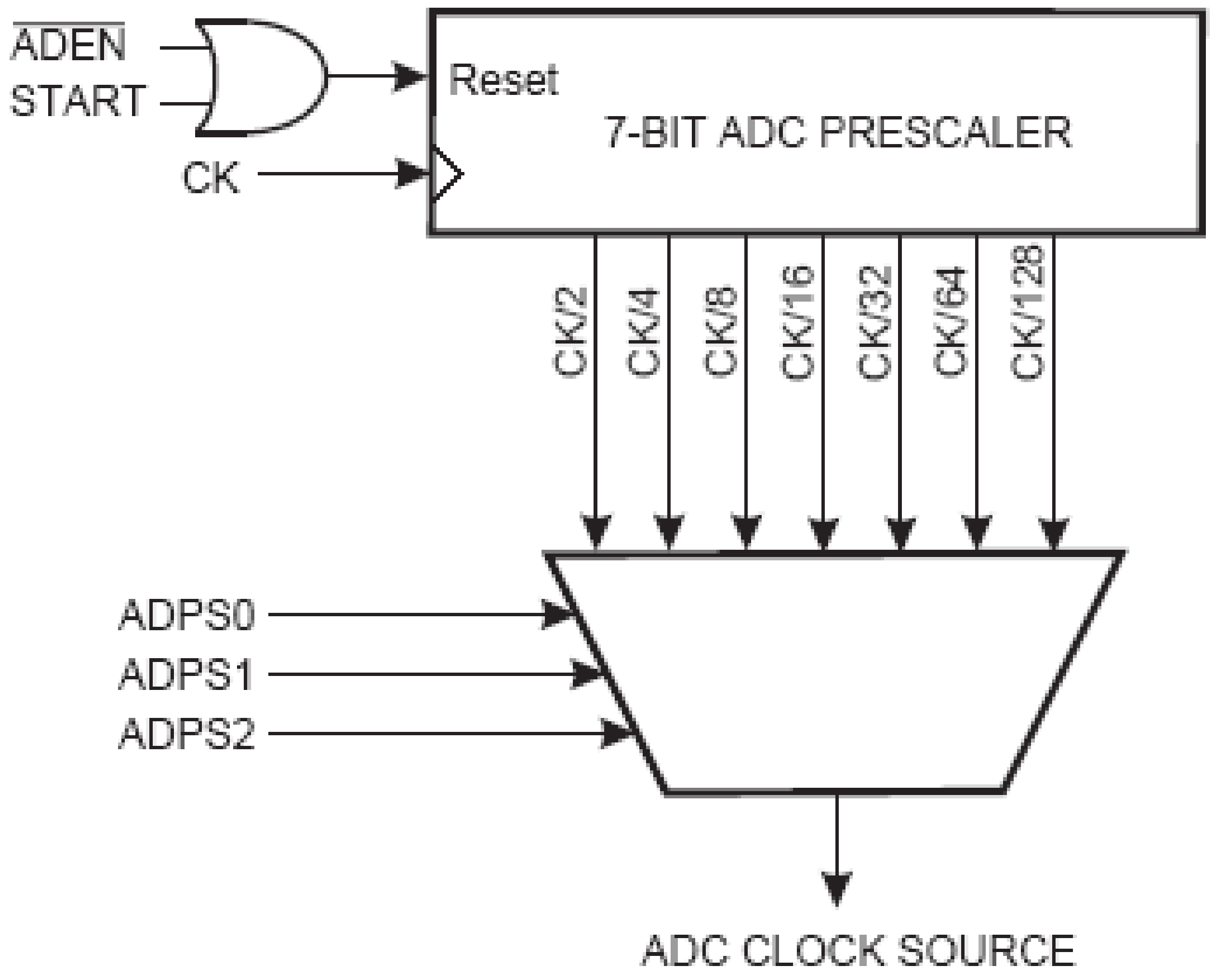
Successive approximation ADC



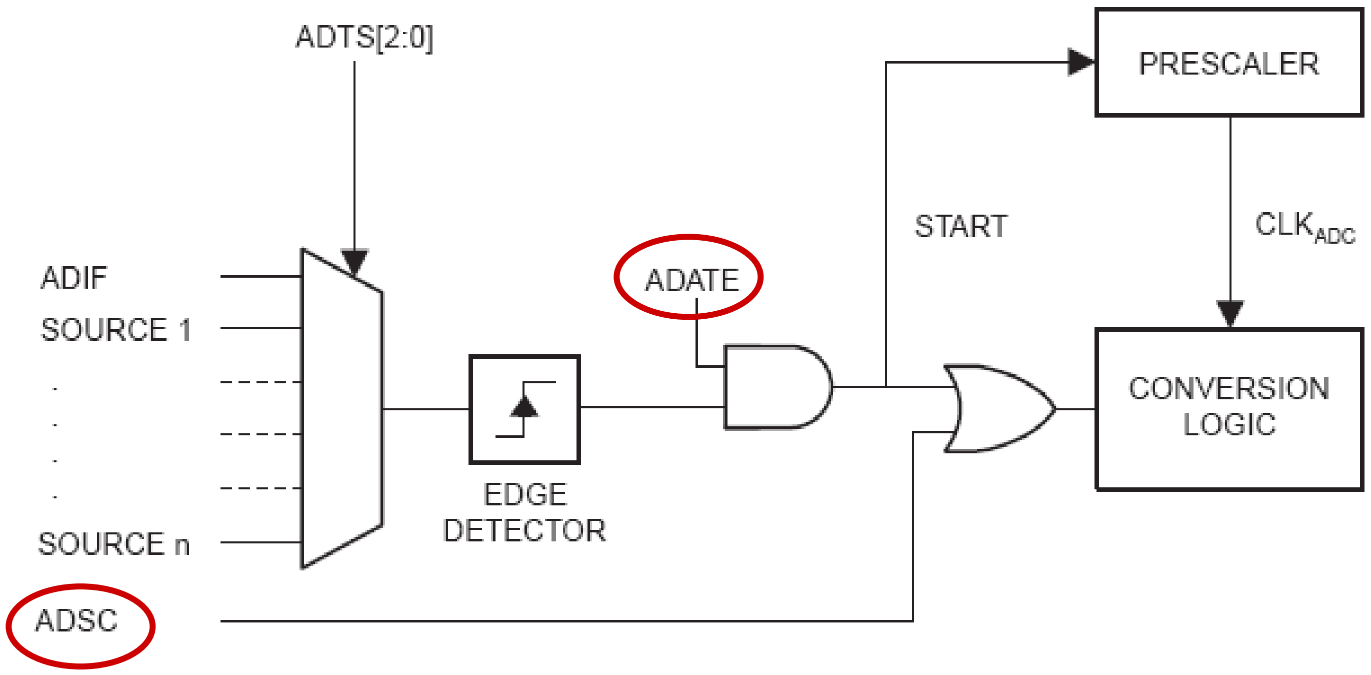
Block diagram, Mega32 ADC



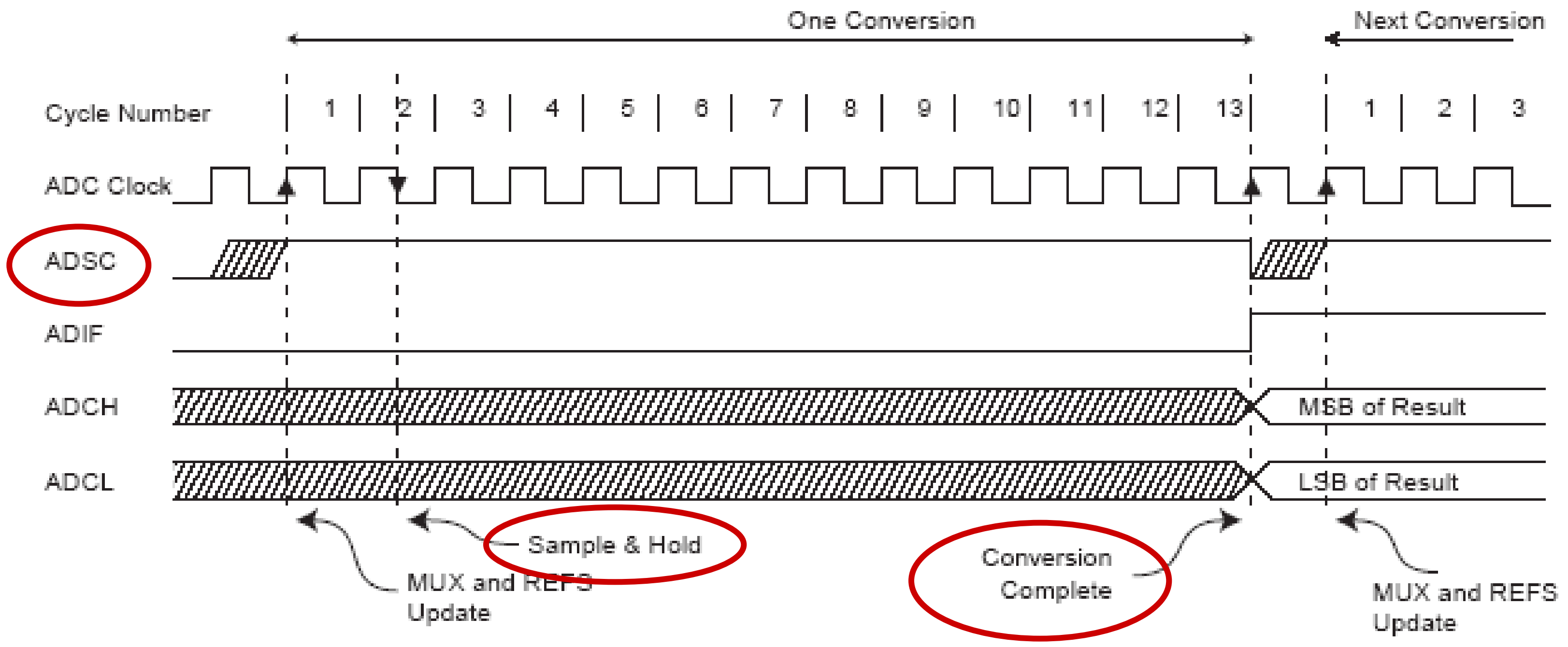
ADC prescaler / clock



ADC start (”trigger”)



ADC timing


One Conversion
Next Conversion
Cycle Number
1
2
3
4
5
6
7
8
9
10
11
12
13
1
2
3
ADC Clock
ADSC
ADIF
ADCH
ADCL
MSB of Result
LSB of Result
Sample & Hold
MUX and REFS Update
Conversion Complete
MUX and REFS Update

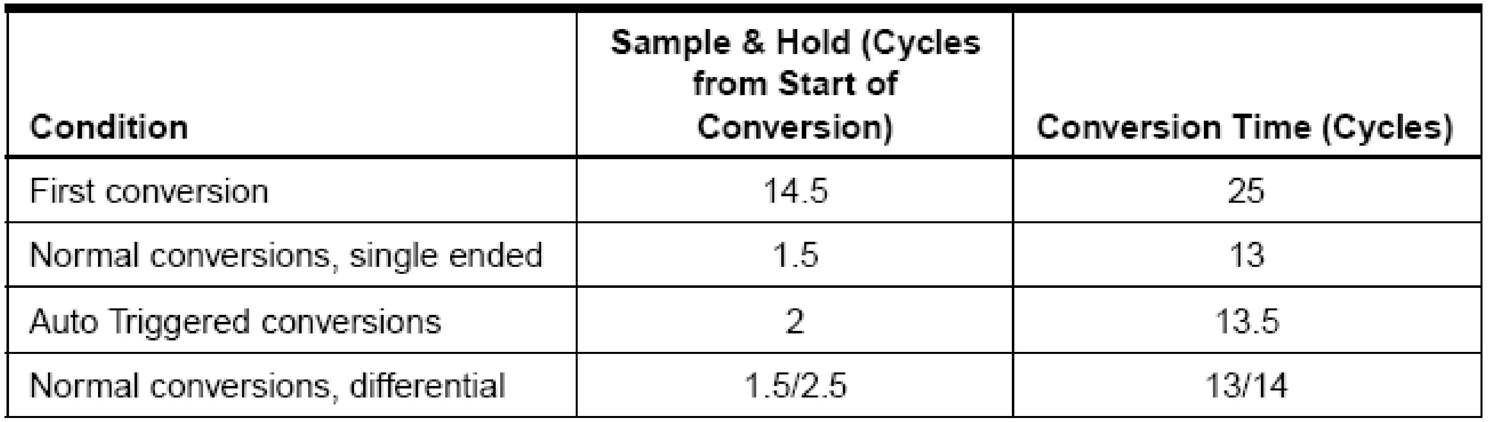
Test (”socrative.com”: Room = MSYS)

- CPU clock frequency = 3,6864 MHz
  ADC prescaler = 64
  Assume 13 ADC clock periods per conversion.

  What is the time for one ADC konversion ?

  A:
  3,5 mikrosekunder
  B:
  226 mikrosekunder
  C:
  17,4 mikrosekunder

Converting times



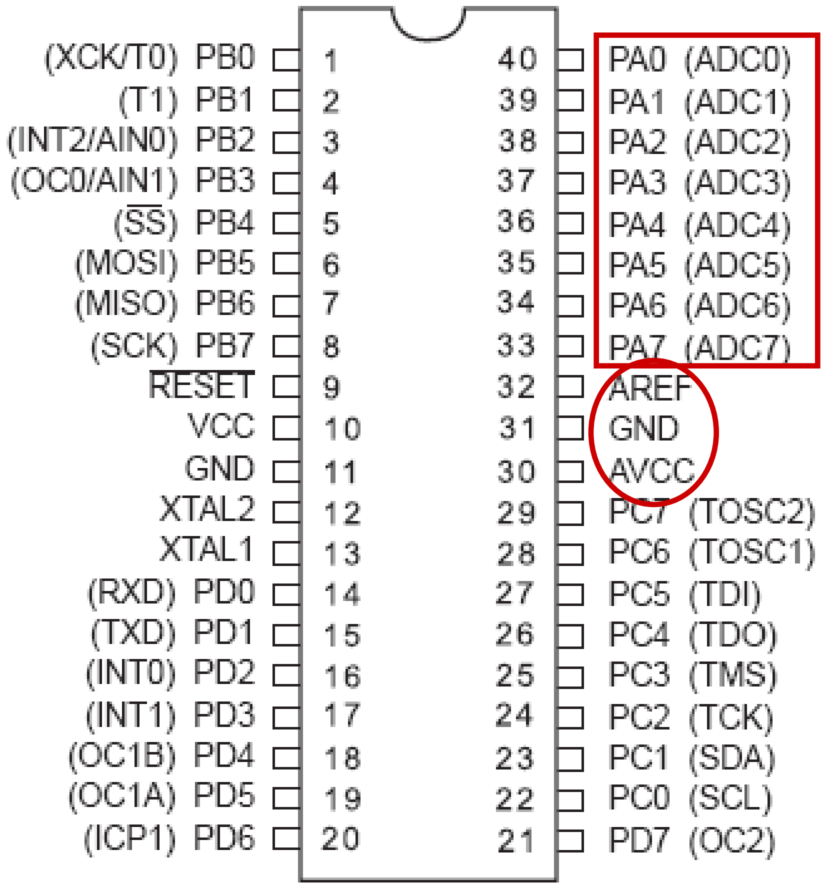
Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
First conversion	14.5	25
Normal conversions, single ended	1.5	13
Auto Triggered conversions	2	13.5
Normal conversions, differential	1.5/2.5	13/14

Example :

f_{ADC} = 200 kHz =>
13 cycles = 65 uS =>

15380 conversions / second

ADC pins


(XCK/T0) PB0 1
(T1) PB1 2
(INT2/AIN0) PB2 3
(OC0/AIN1) PB3 4
(SS) PB4 5
(MOSI) PB5 6
(MISO) PB6 7
(SCK) PB7 8
RESET 9
VCC 10
GND 11
XTAL2 12
XTAL1 13
(RXD) PD0 14
(TXD) PD1 15
(INT0) PD2 16
(INT1) PD3 17
(OC1B) PD4 18
(OC1A) PD5 19
(ICP1) PD6 20
40 PA0 (ADC0)
39 PA1 (ADC1)
38 PA2 (ADC2)
37 PA3 (ADC3)
36 PA4 (ADC4)
35 PA5 (ADC5)
34 PA6 (ADC6)
33 PA7 (ADC7)
32 AREF
31 GND
30 AVCC
29 PC7 (TOSC2)
28 PC6 (TOSC1)
27 PC5 (TDI)
26 PC4 (TDO)
25 PC3 (TMS)
24 PC2 (TCK)
23 PC1 (SDA)
22 PC0 (SCL)
21 PD7 (OC2)

Selecting reference

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
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	

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with

AREF = External pin

AVCC = 5 volt internal

Internal 2.56 volt

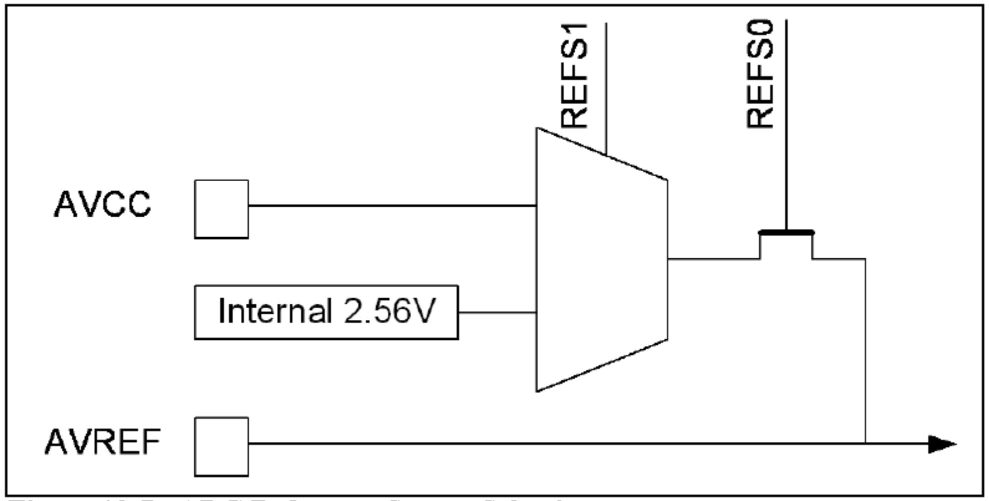


Figure 13-7: ADC Reference Source Selection

Test ("socrative.com": Room = MSYS)

- We are using the ADC for measuring a constant DC voltage.
  The ADC output is 400, when the voltage is 2 volt.
  What will be the ADC output, if the voltage is changed to 1 volt ?
  A:
    800
  B:
    400
  C:
    200
- D:
    100

Test (”socrative.com”: Room = MSYS)

- We are using the ADC for measuring a constant DC voltage.
  The ADC output is 400, when the reference voltage is 4 volt.
  What will be the ADC output, if the reference voltage is changed to 2 volt ?
  A:
    200
  B:
    300
  C:
    400
- D:
    800

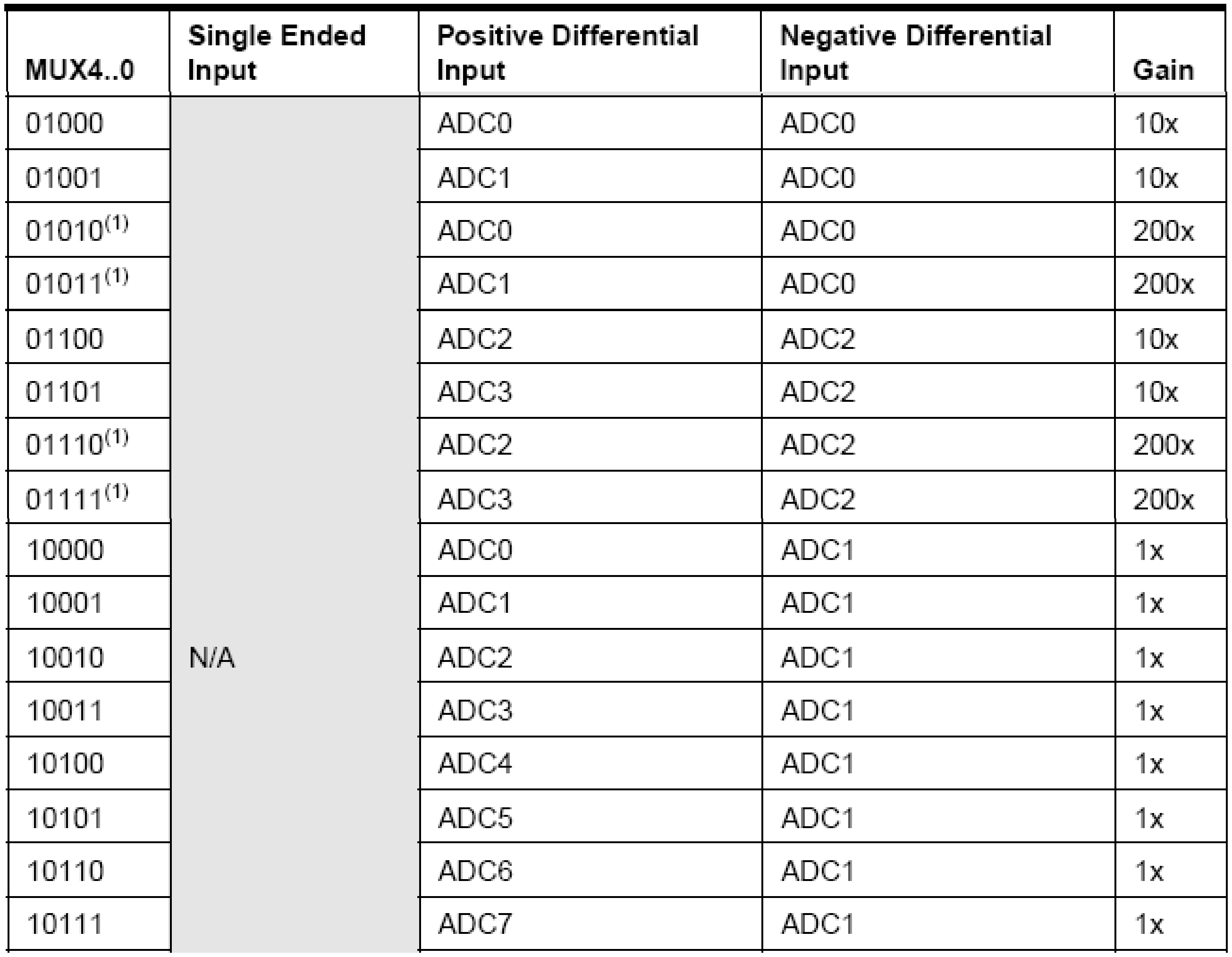
Selecting ADC input (1)

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
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	

MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			

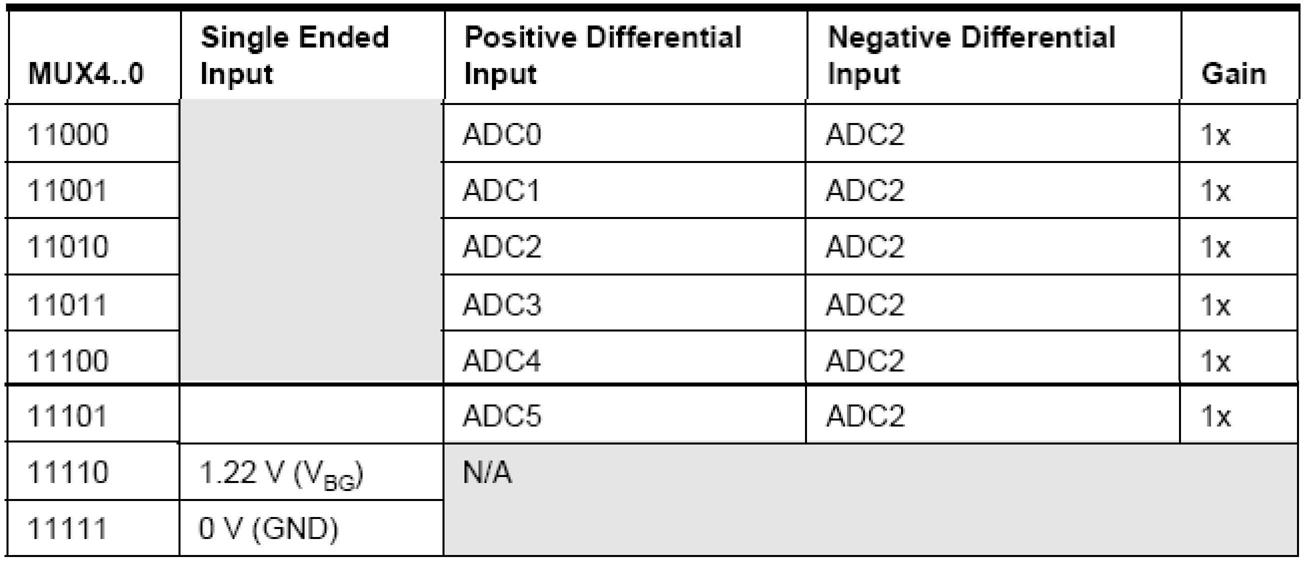
"Single ended" = <u>One</u> voltage referenced to ground.

Selecting ADC input (2)



MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
01000		ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010[(1)]		ADC0	ADC0	200x
01011[(1)]		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110[(1)]		ADC2	ADC2	200x
01111[(1)]		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010	N/A	ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x

Selecting ADC input (3)



MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x
11101		ADC5	ADC2	1x
11110	1.22 V (V_{BG})	N/A		
11111	0 V (GND)			

Selecting the ADC in

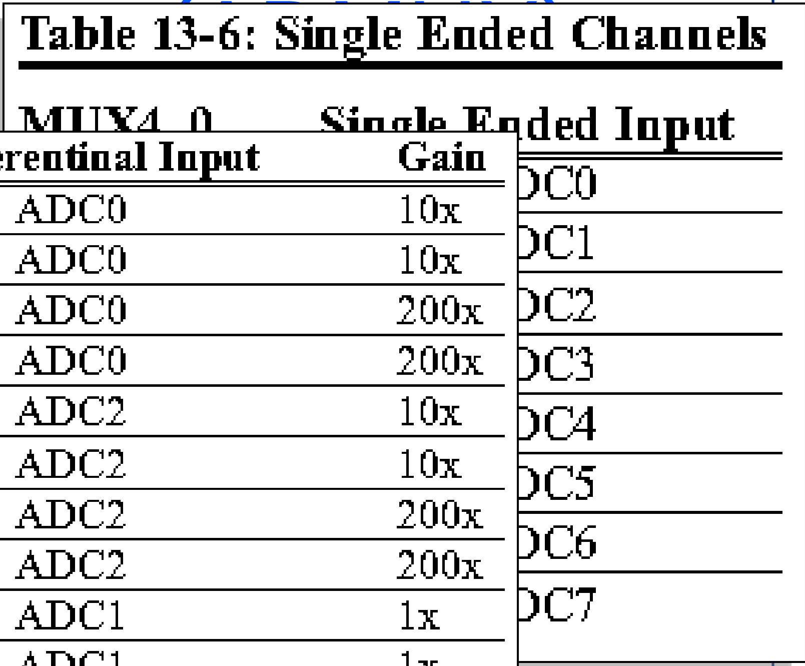
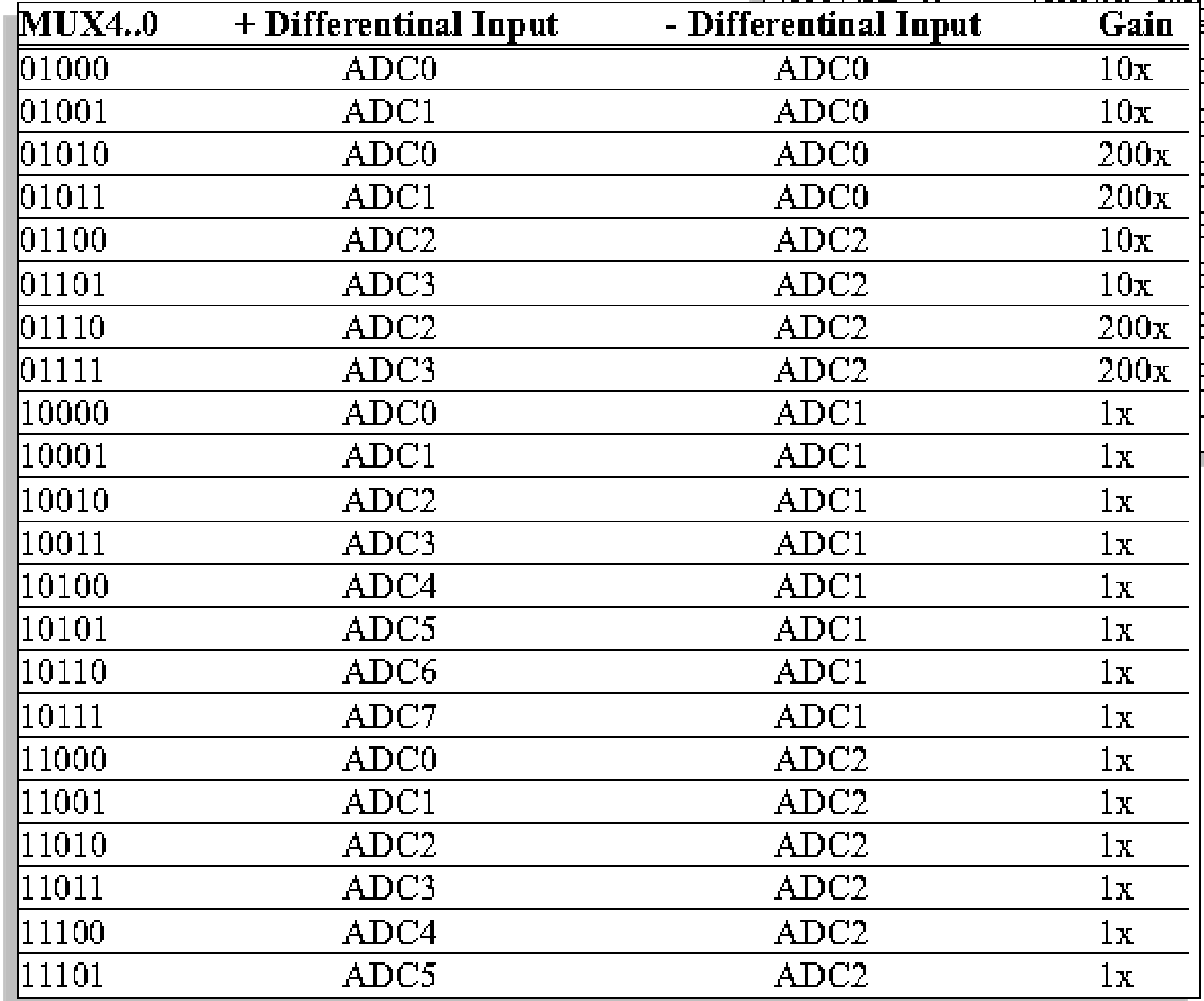


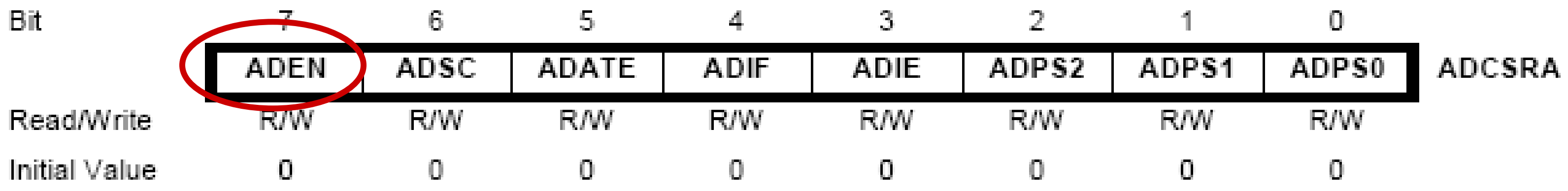
Table 13-6: Single Ended Channels

MUX4..0	Single Ended Input
	DC0
	DC1
	DC2
	DC3
	DC4
	DC5
	DC6
	DC7



MUX4..0	+ Differential Input	- Differential Input	Gain
01000	ADC0	ADC0	10x
01001	ADC1	ADC0	10x
01010	ADC0	ADC0	200x
01011	ADC1	ADC0	200x
01100	ADC2	ADC2	10x
01101	ADC3	ADC2	10x
01110	ADC2	ADC2	200x
01111	ADC3	ADC2	200x
10000	ADC0	ADC1	1x
10001	ADC1	ADC1	1x
10010	ADC2	ADC1	1x
10011	ADC3	ADC1	1x
10100	ADC4	ADC1	1x
10101	ADC5	ADC1	1x
10110	ADC6	ADC1	1x
10111	ADC7	ADC1	1x
11000	ADC0	ADC2	1x
11001	ADC1	ADC2	1x
11010	ADC2	ADC2	1x
11011	ADC3	ADC2	1x
11100	ADC4	ADC2	1x
11101	ADC5	ADC2	1x

ADC enable



Bit	7	6	5	4	3	2	1	0	
	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	

Before we can use the ADC, it has to be enabled (write 1 to ADCSRA bit 7).

This ”turns on” the ADC hardware.

ADC prescaler

Bit	7	6	5	4	3	2	1	0	
	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	

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

NOTICE : The ADC clock must be between 50 kHz and 200 kHz !
(To obtain 10 bits of accuracy).

Test (”socrative.com”: Room = MSYS)

- The CPU clock frequency is 12 MHz.
  We want the ADC clock frequency to be between 50 kHz and 200 kHz (for best performance).

  What ADCSRA value is WRONG ?

  A:
  ADCSRA = 0b10000111;
  B:
  ADCSRA = 0b10000110;
  C:
  ADCSRA = 0b10000101;

Normal / ”Left Adjust Result”

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
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	

ADLAR = 0

ADCL MUST be read FIRST.
The ADCH MUST be read.

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

Normally only ADCH is read.

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	

ADCH and ADCL Data registers

- ADCH:ADCL store the results of conversion.
- The 10 bit result can be right or left justified:

Our compiler:
ADCW is ADCH—ADCL:
x = ADCW;

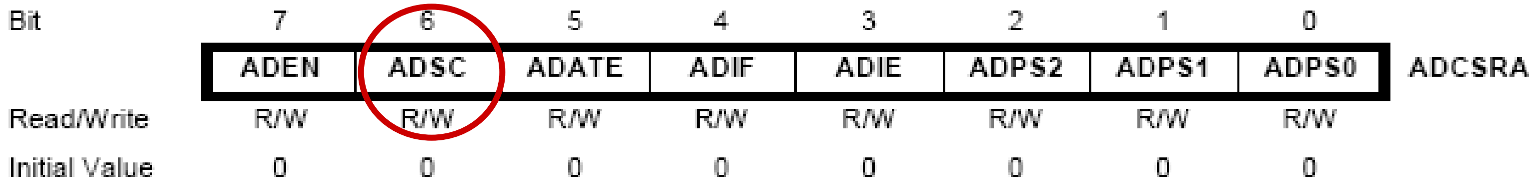
ADLAR = 0

ADCH								ADCL							
-	-	-	-	-	-	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0

ADLAR =1

ADCH								ADCL							
ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	-	-	-	-	-	-

Manual START



Bit	7	6	5	4	3	2	1	0	
	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 ADSC = 1 : Conversion starts

  ```
  ADCSRA |= 0b01000000;
  ```

- Bit ADSC == 0 : Conversion ended

  ```
  while (ADCSRA & 0b01000000)
  { }
  // Now ADC can be read
  x = ADCW;
  ```

Automatic START ("trigger")

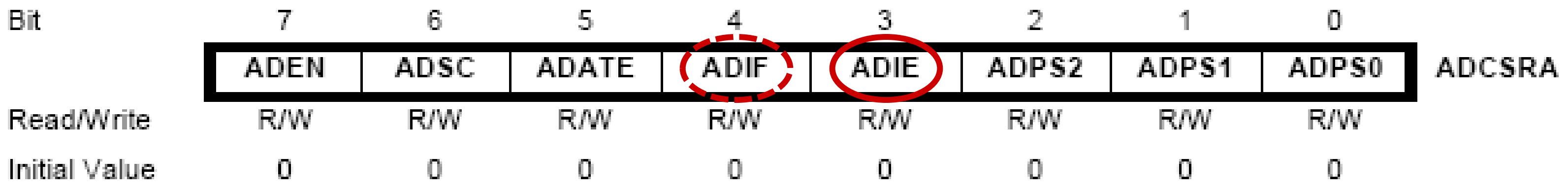
Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

+

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	–	ACME	PUD	PSR2	PSR10	SFIOR
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	

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
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

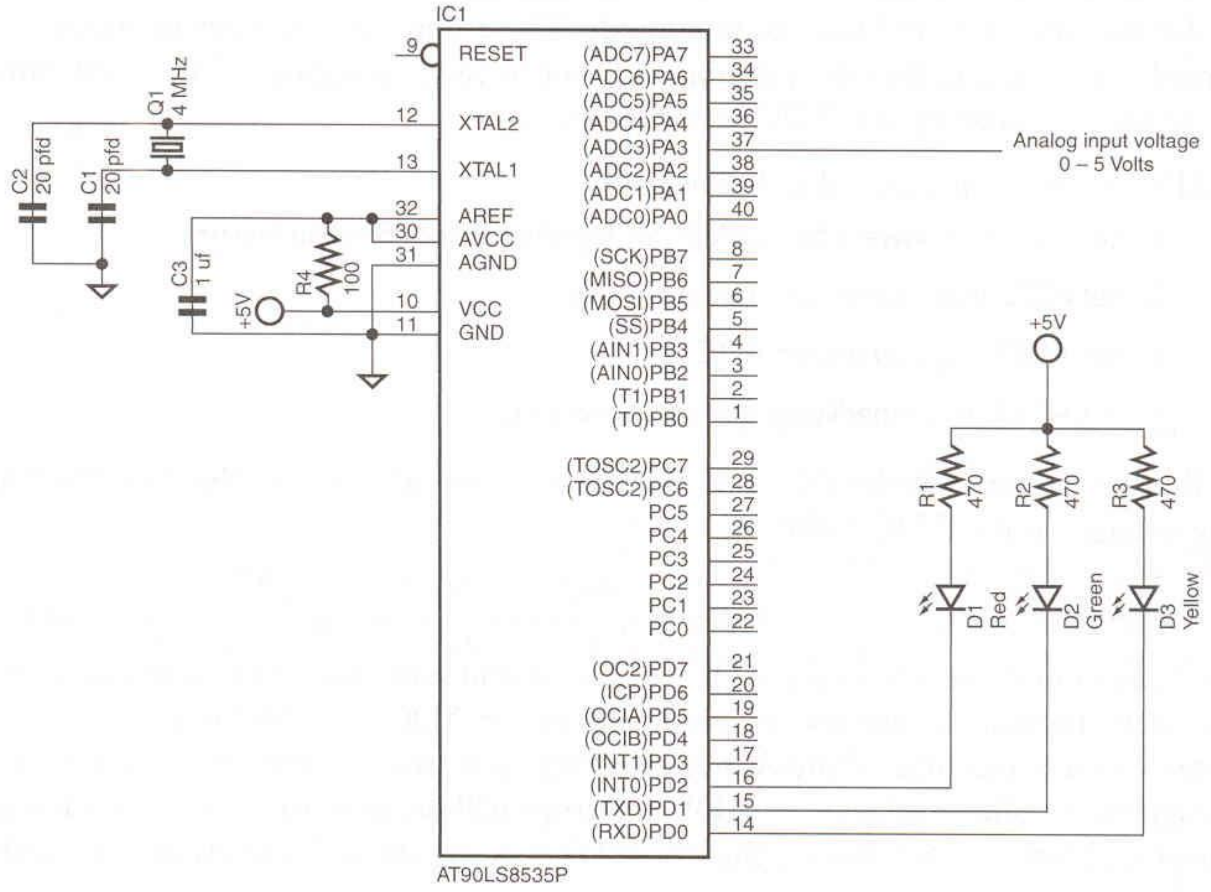
ADC interrupt



Bit	7	6	5	4	3	2	1	0	
	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	

- ADIE: ”ADC interrupt enable”.
  1 => The ADC interrupt is generated if the global interrupt enable flag is also set.
- (ADIF: ”ADC interrupt flag”)
  Is set high following each conversion.
  * Cleared automatically in the interrupt routine
  OR
  * by writing a 1 to bit ADIF.

Example 1: Level tester



End of lesson 19