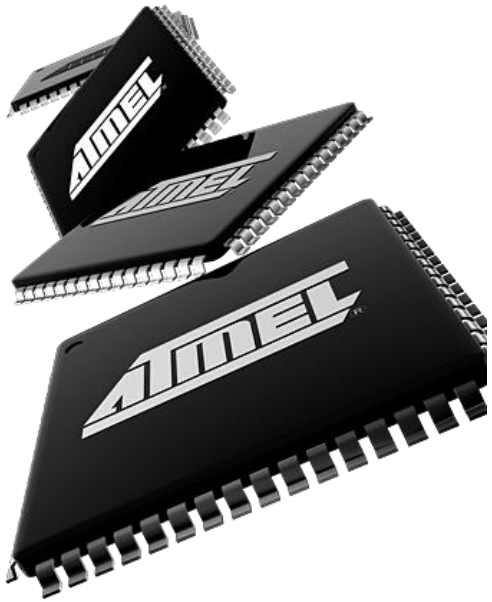
Principles and Applications of Microcontrollers

Yan-Fu Kuo
Dept. of Biomechatronics Engineering
National Taiwan University

Today:

- Analog-to-digital converter (ADC)

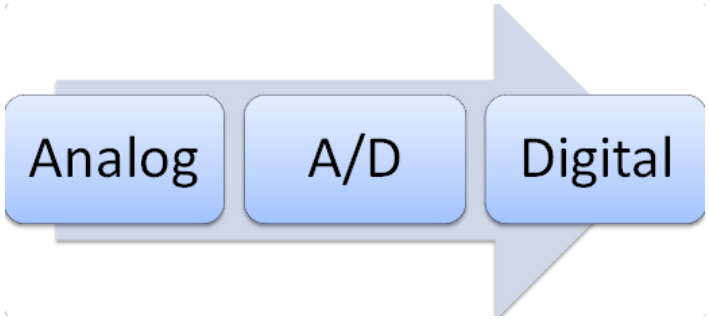


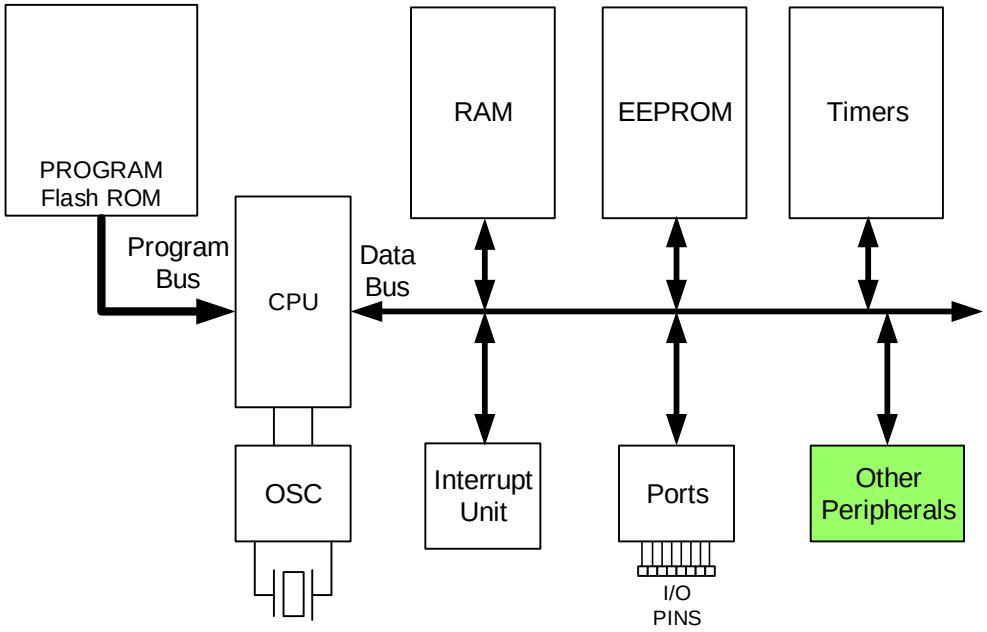
Review – Bit-wise Operation

- Check a bit in a byte
- Write one to a bit in a byte without changing its content
- Write one to two bits in a byte
- Write zero to a bit in a byte

Outline

- Analog-to-digital converter (ADC)
  - Analog and digital signals
  - Successive approximation ADC
  - AVR ADC connection
  - ADC registers
  - Single conversion mode
  - Free running mode
- Getting started



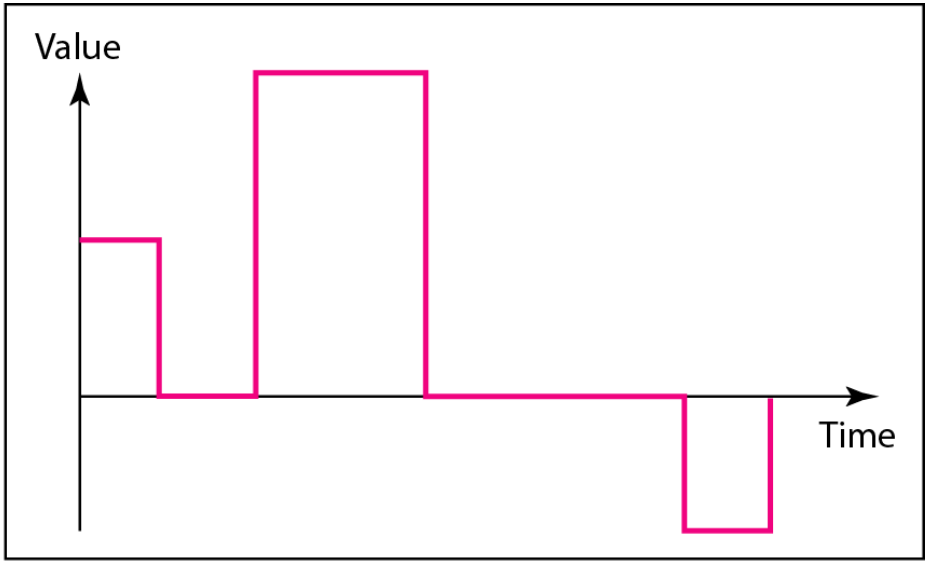


Analog vs. Digital Signal

Analog vs. Digital Signal

- Analog signal – directly measurable quantities that are continuous both in magnitude and time
- Digital signal – measurement in states, e.g., binary 0 and 1, that are discrete both in magnitude and time

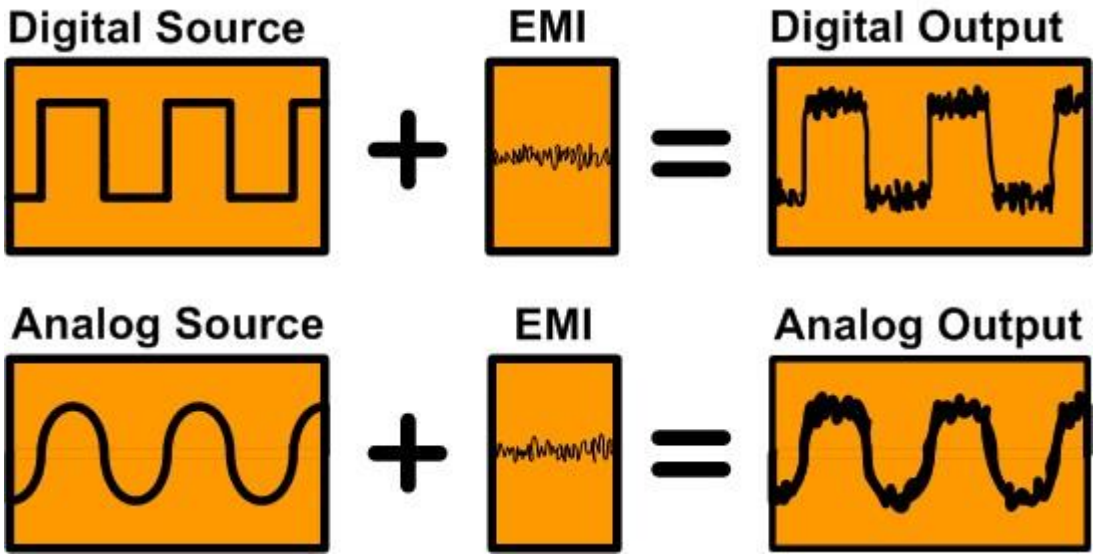
a. Analog signal



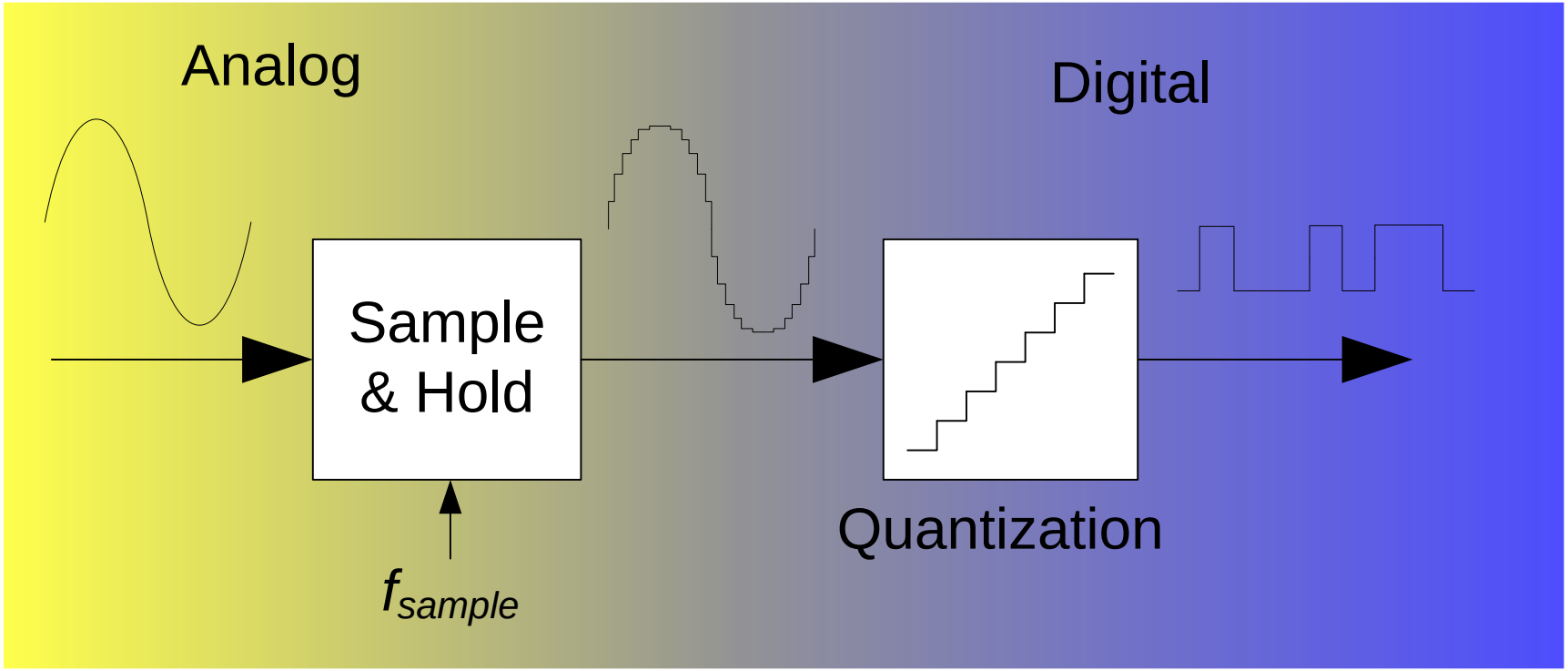
b. Digital signal

Advantage of Digital Signal

- Increased noise immunity
- Easy to compute nonlinear functions
- Reliable and reproducible



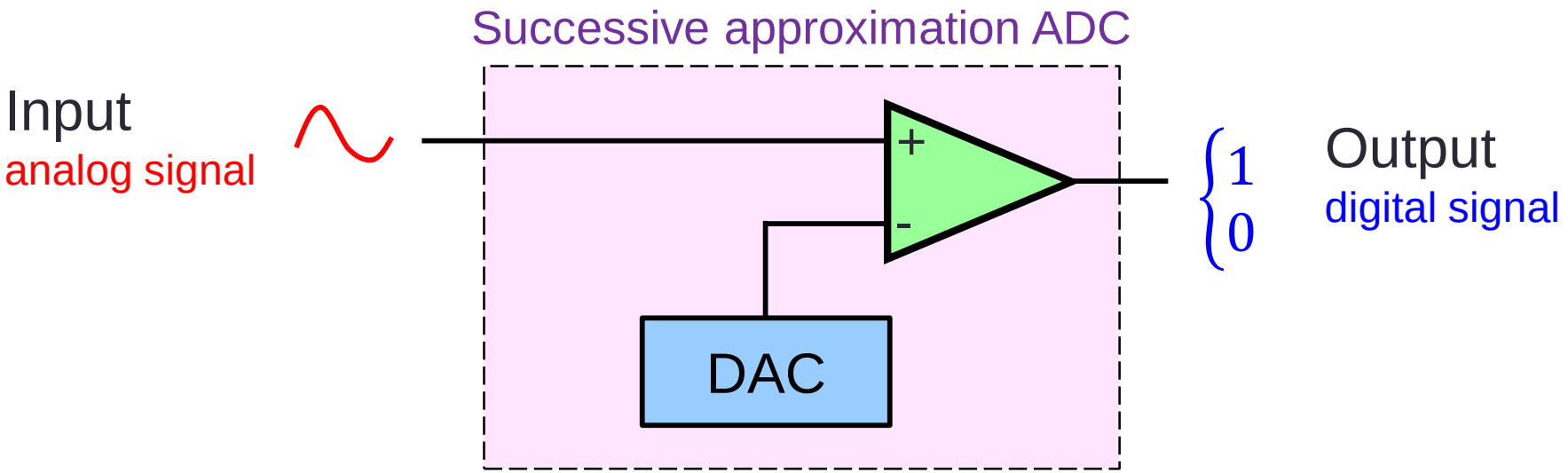
Scheme of Analog-to-digital Conversion



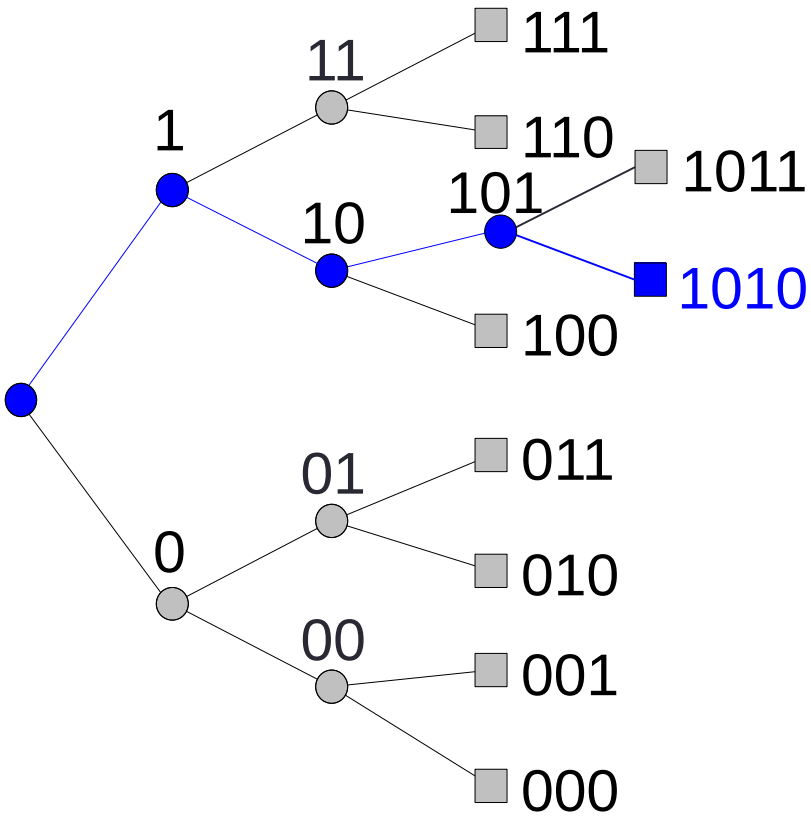
- Quantization – discretizing input magnitude values
- Sampling – reduction of a continuous signal to a discrete signal

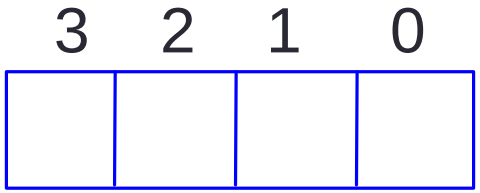
AVR ADC

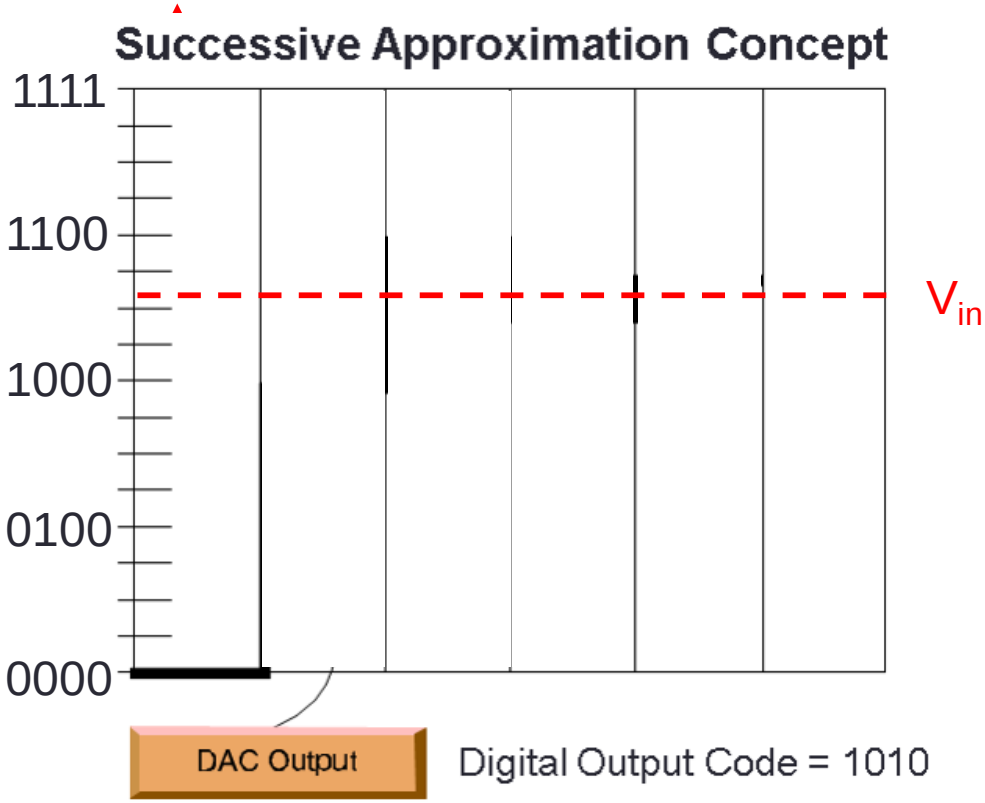
- 6 input channels
- 10-bit resolution
- ±2 LSB absolute accuracy
- Successive approximation ADC
- 13 - 260μs conversion time



Successive Approximation ADC

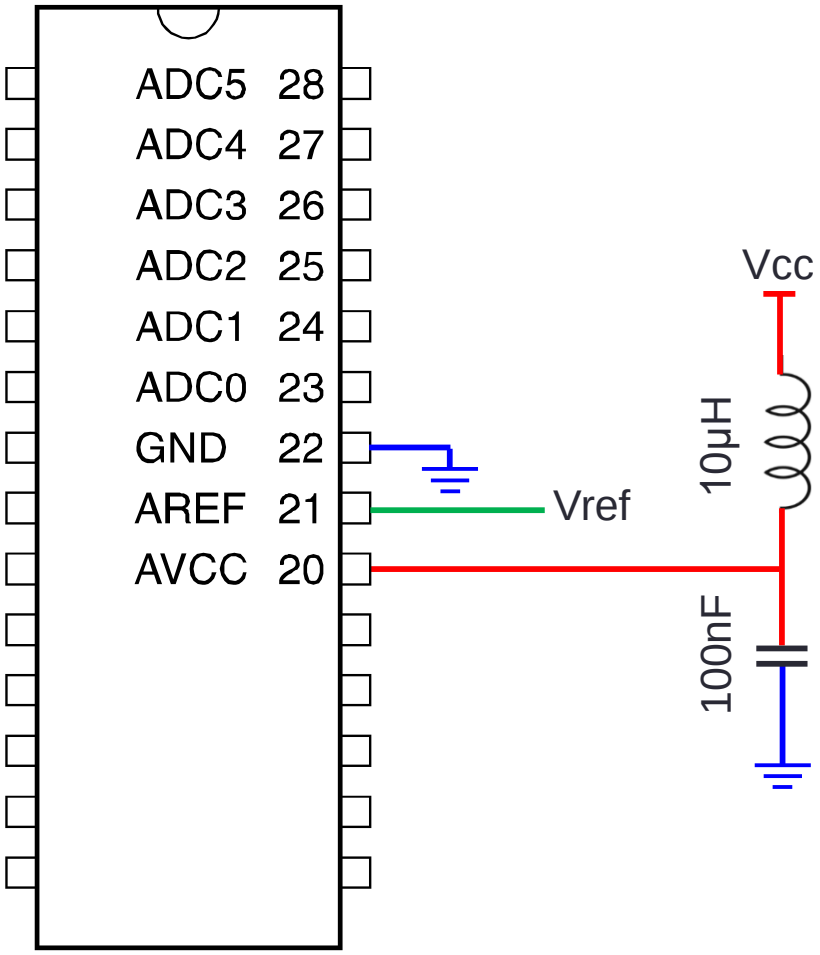






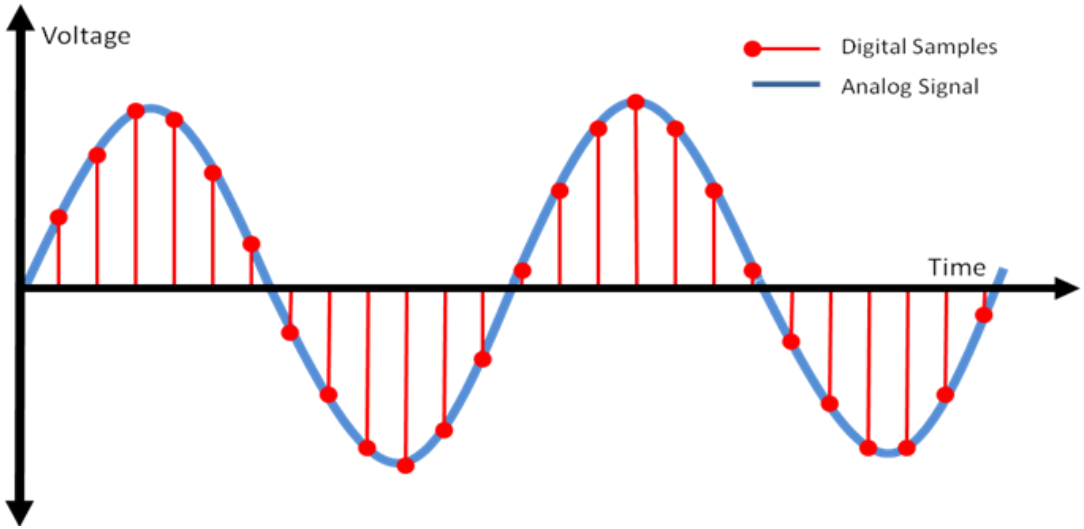
ADC Connection

- Six analog input pins: PC0 – PC5
- AVcc is the power supply to ADC
- Suggested connection for noise cancelation
- ADC range: 0 – Vref
- Internal Vref available



Outline (Cont'd)

- Analog-to-digital converter (ADC)
  - Analog and digital signals
  - Successive approximation ADC
  - AVR ADC connection
  - ADC registers
  - Single conversion mode
  - Free running mode
- Getting started



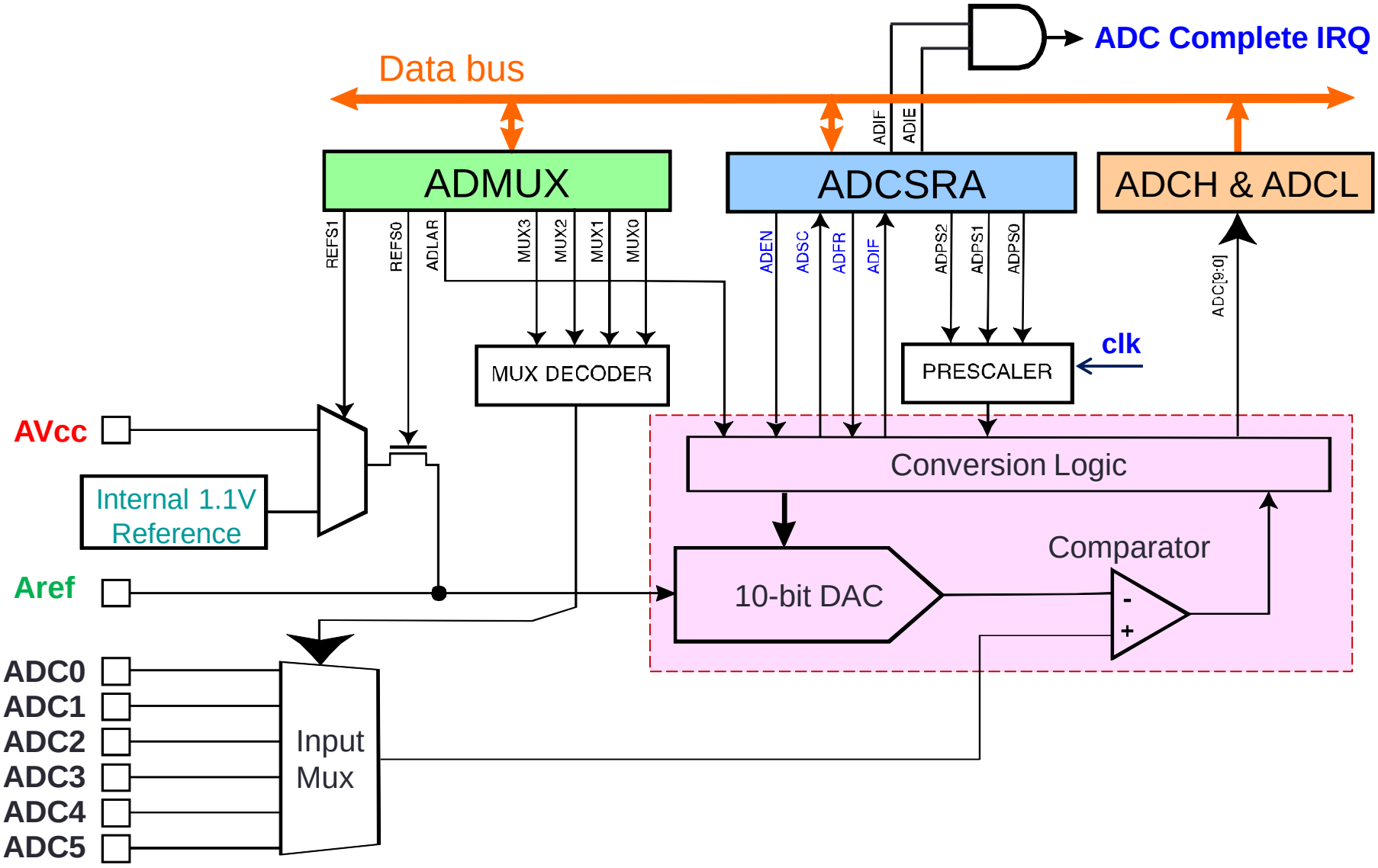
ADC Registers

ADMUX
- Input pin and reference voltage control

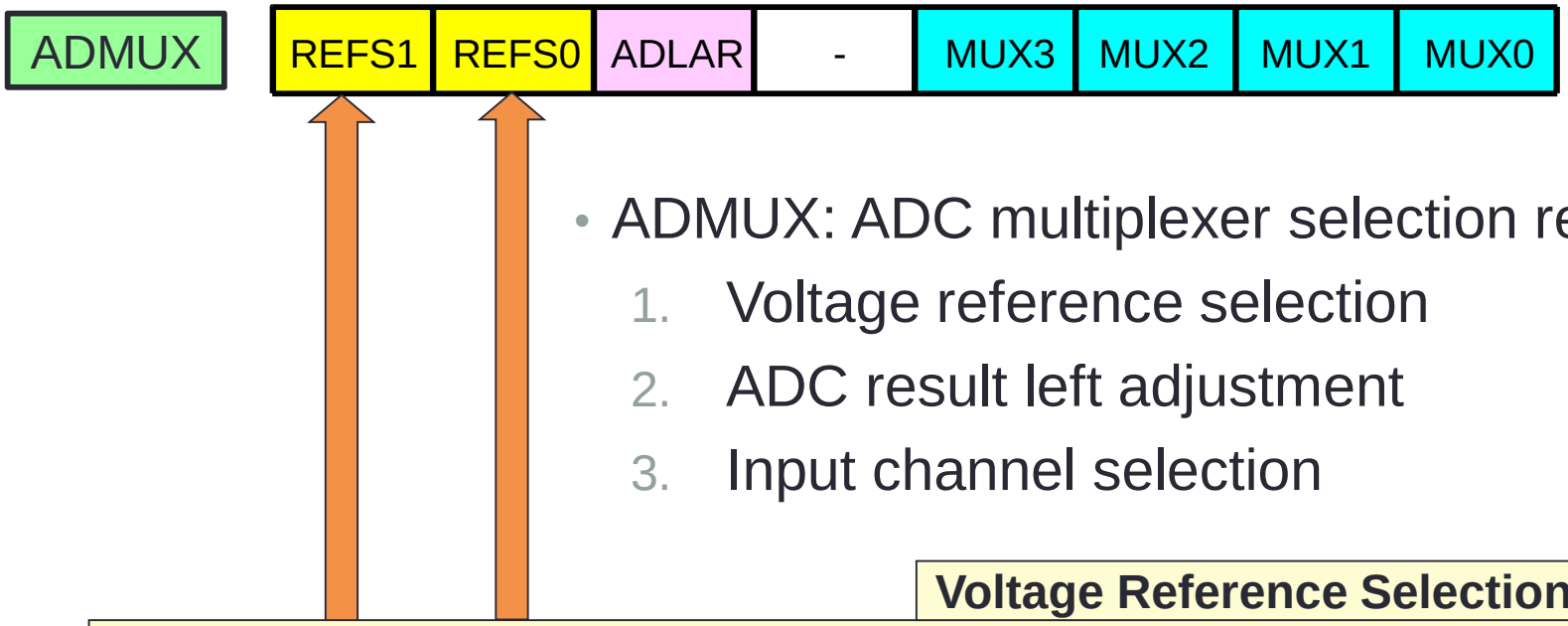
ADCSRA
- Status and prescalar control

ADCH
ADCL
- ADC data storage
- 10-bit ADC needs 2 bytes for storing

ADC Schematic Diagram



Voltage Reference Selection

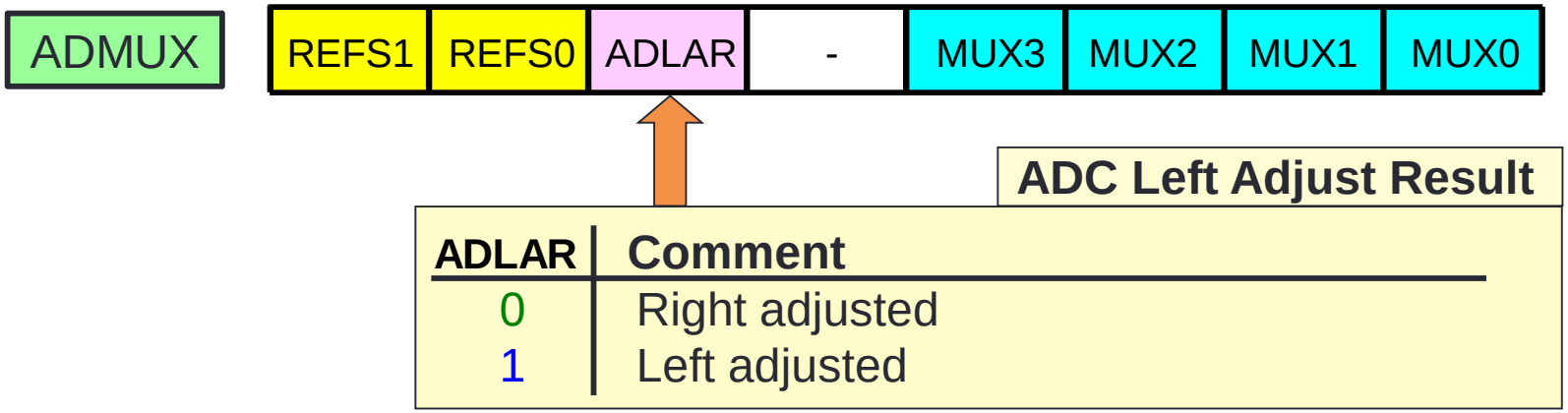


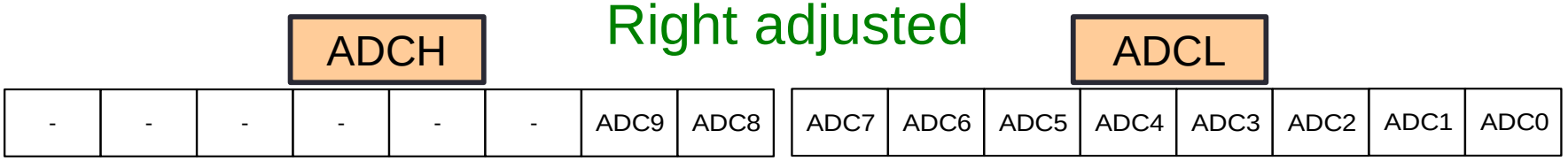
- ADMUX: ADC multiplexer selection register
  1. Voltage reference selection
  2. ADC result left adjustment
  3. Input channel selection

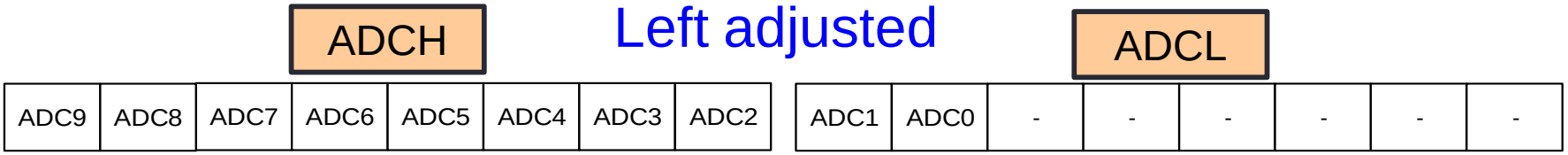
Voltage Reference Selection

REFS1	REFS0	Comment
0	0	Aref
0	1	Avcc
1	0	Reserved
1	1	Internal 1.1V

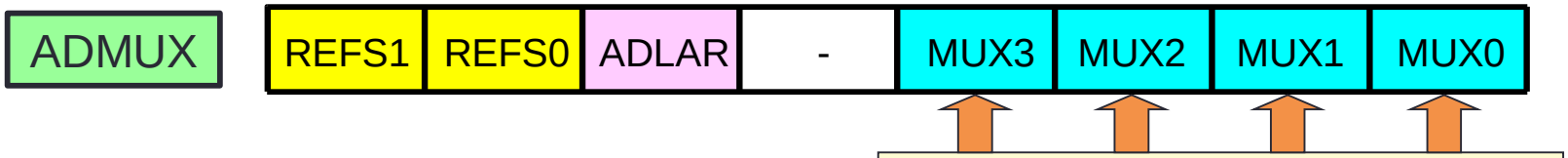
ADC Result Left Adjustment







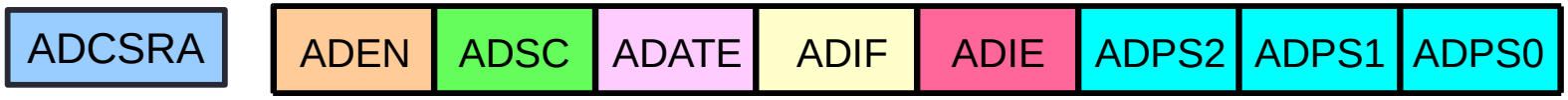
Input Channel Selection



Input Channel Selection

MUX3	MUX2	MUX1	MUX0	Comment
0	0	0	0	ADC0
0	0	0	1	ADC1
0	0	1	0	ADC2
0	0	1	1	ADC3
0	1	0	0	ADC4
0	1	0	1	ADC5
1	1	1	1	0V

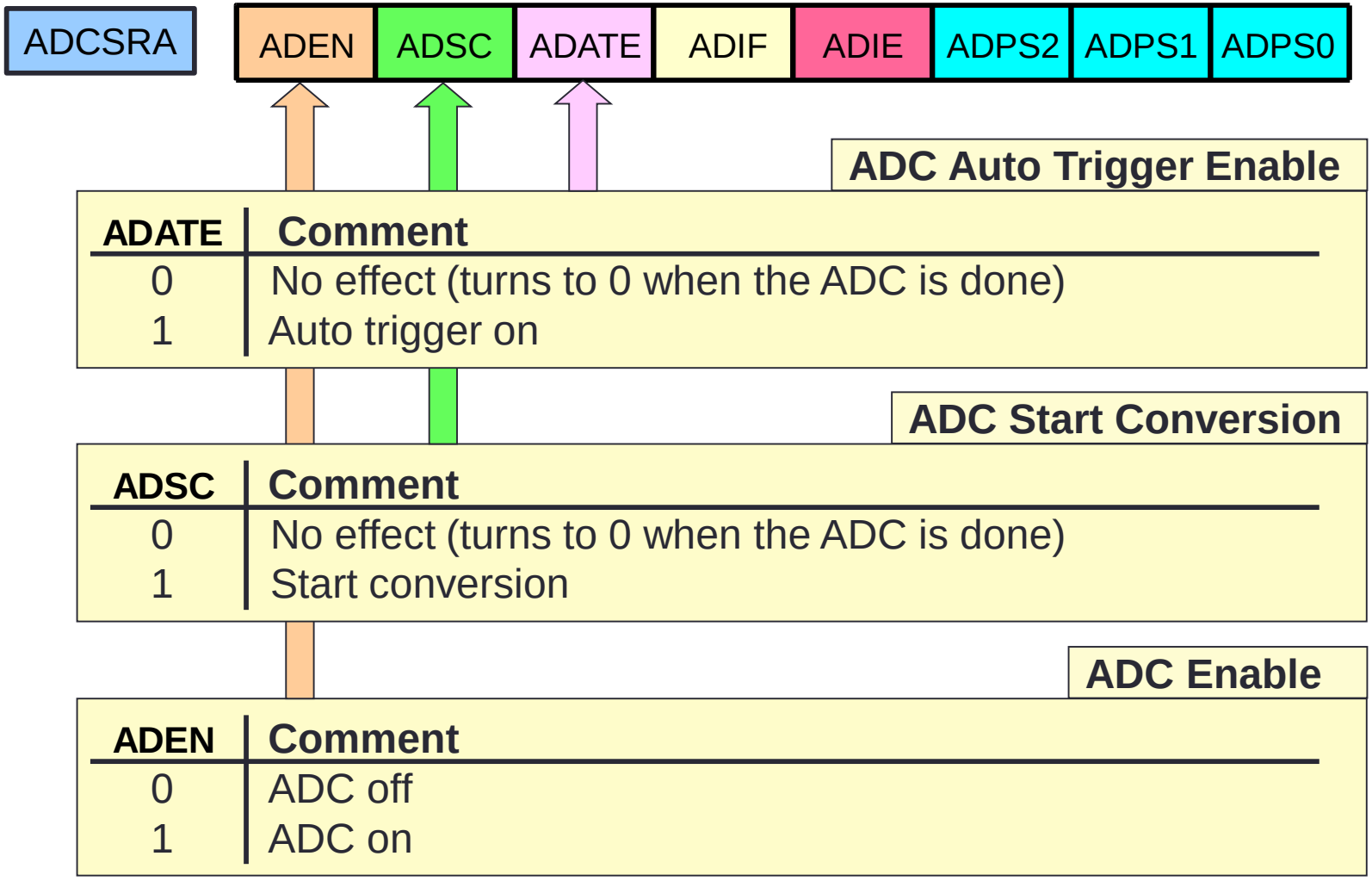
ADC Control and Status Register A



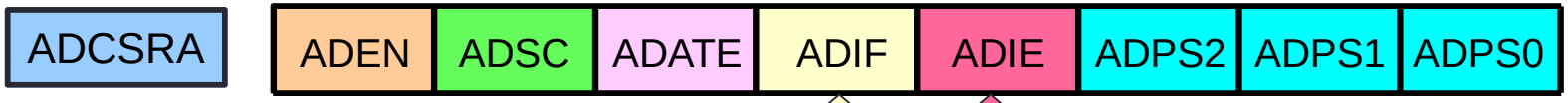
- ADCSRA: ADC control and status register A
  1. ADC enable
  2. ADC start conversion
  3. ADC auto trigger enable
  4. ADC interrupt flag
  5. ADC prescaler selection

Note: ADIF is set whenever ADC conversion is done

ADC Enable, Start Conversion, and Auto Trigger



ADC Interrupt Flag and Interrupt Enable



- Interrupt flag (ADIF) is 1 whenever a conversion is done

ADC Interrupt Enable

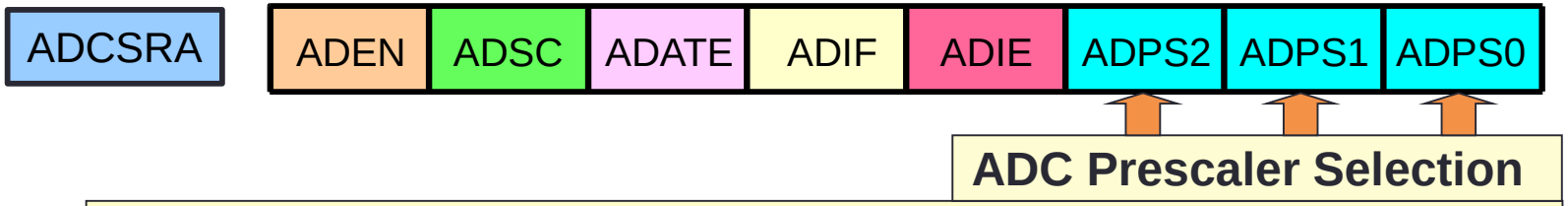
ADIE	Comment
0	Disable interrupt
1	Enable interrupt

ADC Interrupt Flag

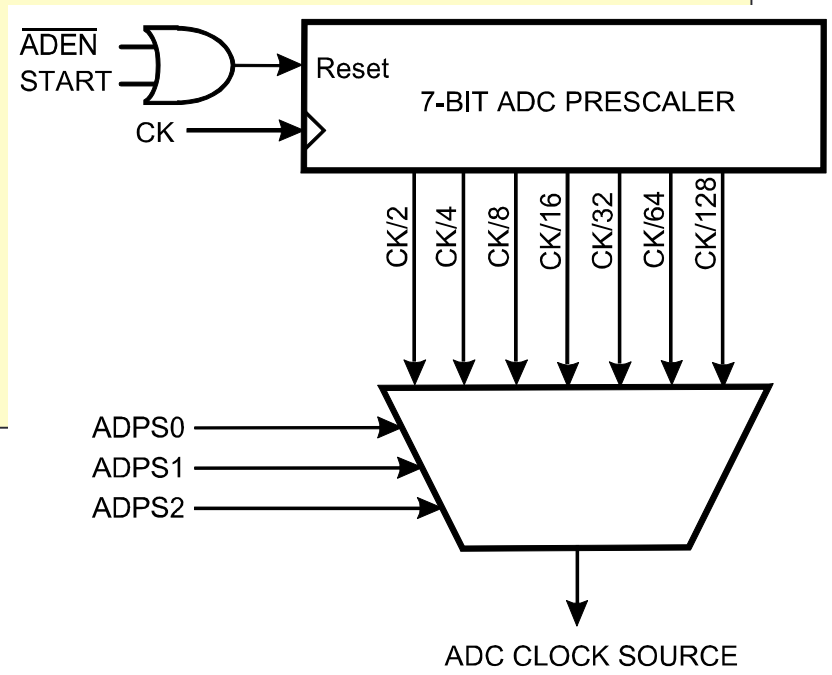
ADIF	Comment
0	No effect
1	Clear the flag

Note: ADIF can be cleared by written 1 into the bit

ADC Prescaler Selection



ADPS2	ADPS1	ADPS0	Comment
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



Outline (Cont'd)

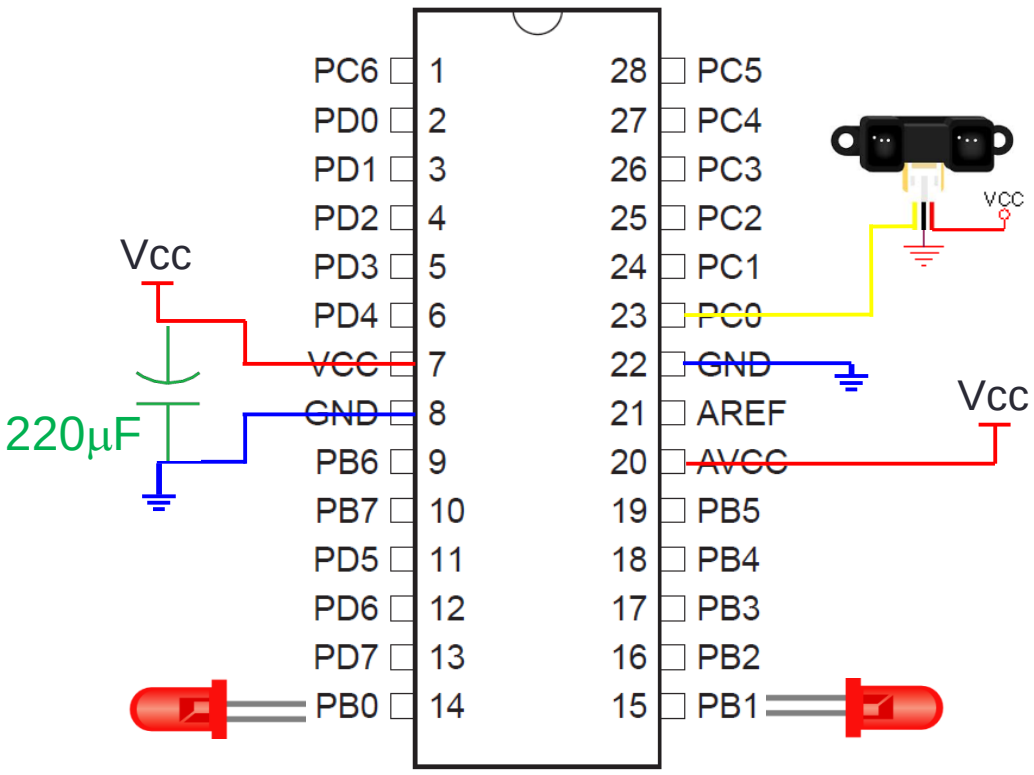
- Analog-to-digital converter (ADC)
  - Analog and digital signals
  - Successive approximation ADC
  - AVR ADC connection
  - ADC registers
  - Single conversion mode
  - Free running mode
- Getting started

Example: Read DMS Sensor

- Read from ADC0 (PC0)
- Right adjusted:

  display the result on
  Port B for the high byte
  Port D for the low byte
- ADC prescalar 128
- Vref = internal 1.1V
- Delay for 200ms

9	8	7	6	5	4	3	2	1	0
PB 1	PB 0	PD 7	PD 6	PD 5	PD 4	PD 3	PD 2	PD 1	PD 0



Flowchart

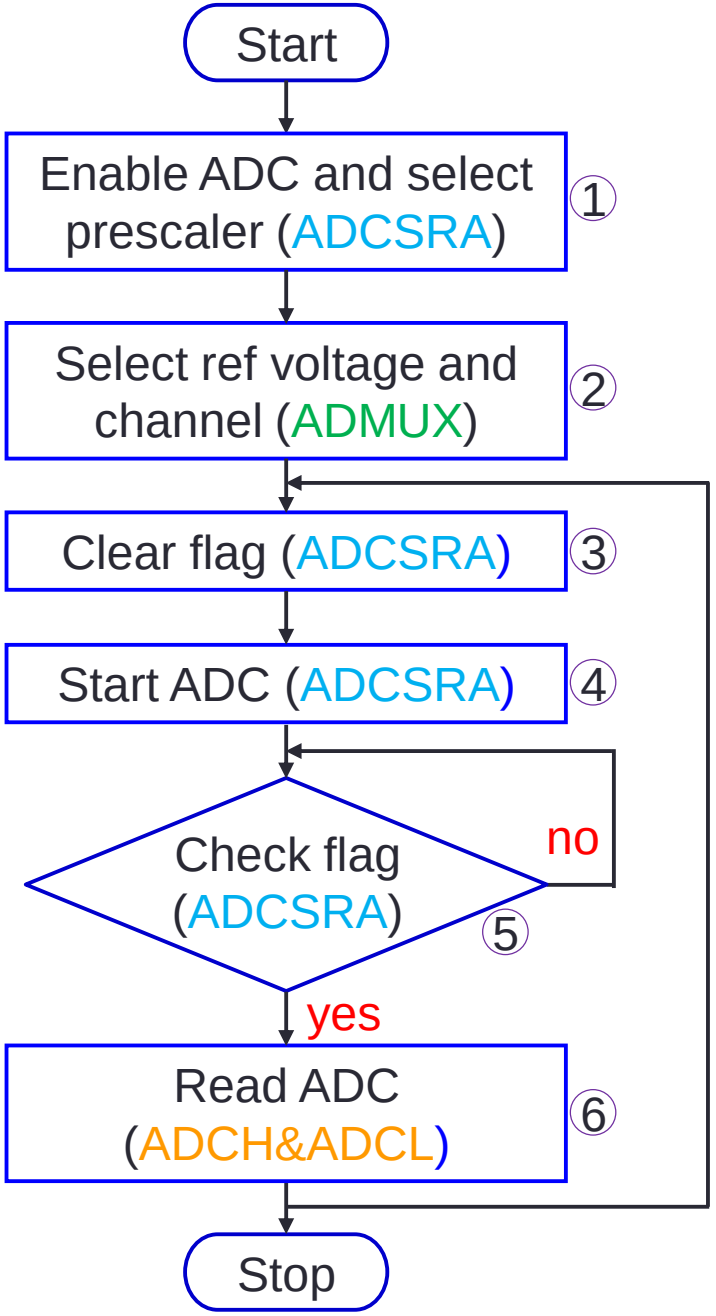
- What value do we set the registers?

ADCSRA

1					1	1	1
ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0

REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0
1	1	0		0	0	0	0

ADMUX



NOTE: read ADCL first!

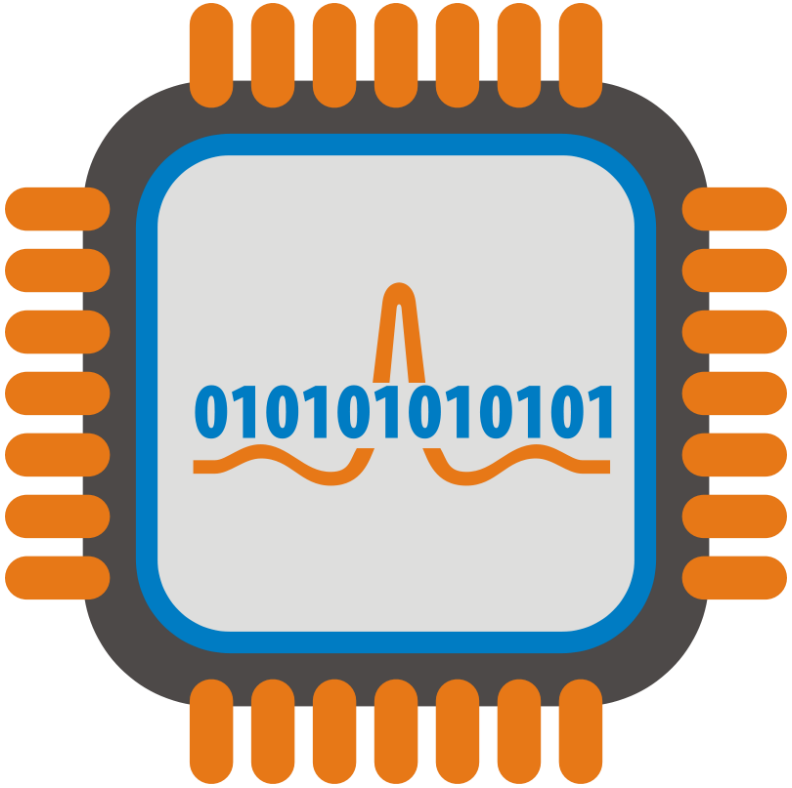
Read DMS Sensor

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
    CLKPR=(1<<CLKPCE);
    CLKPR=0b00000011;                    // set clk to 1Mhz
    DDRB=0xFF;                           // PORTB as output
    DDRD=0xFF;                           // PORTD as output
    DDRC=0;                              // PORTC as input
①   ADCSRA=0b10000111;                   // enable + prescaler
②   ADMUX=0b11000000;                    // ref volt + channel
    while (1) {
③       ADCSRA|=(1<<ADIF);               // clear ADIF
④       ADCSRA|=(1<<ADSC);               // start ADC
⑤       while((ADCSRA&(1<<ADIF))==0);    // wait for ADC done
⑥       PORTD=ADCL;                      // read low byte first
        PORTB=ADCH;
        _delay_ms(200);
    }
}
```

Outline (Cont'd)

- Analog-to-digital converter (ADC)
  - Analog and digital signals
  - Successive approximation ADC
  - AVR ADC connection
  - ADC registers
  - Single conversion mode
  - Free running mode
- Getting started

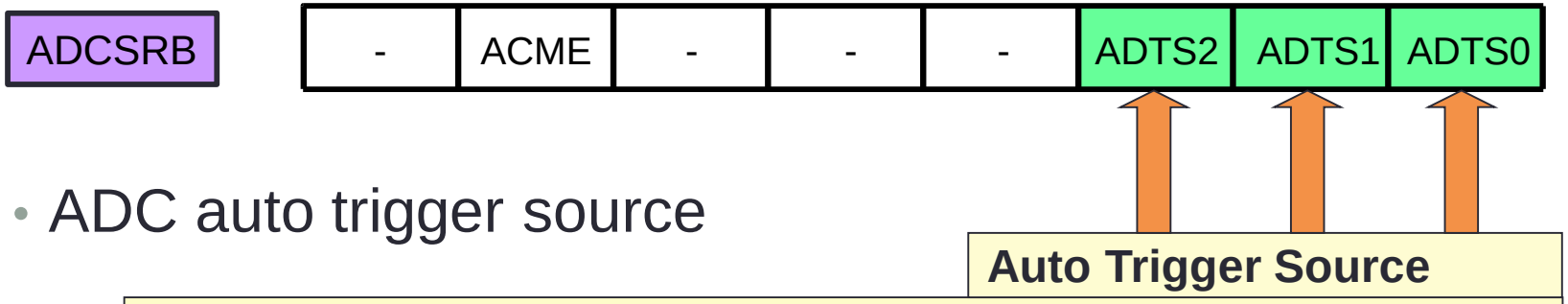


ADC Running Mode

- Single conversion mode
  - Triggered by your program
  - Write 1 into ADSC in ADCSRA

- Auto trigger mode
  - Triggered automatically or by some events
  - Write 1 into ADATE in ADCSRA
  - Setup the trigger source in ADCSRB
    1. Free running mode
    2. External interrupt request
    3. Timer compare match
    4. Timer overflow

ADC Control and Status Register B



- ADC auto trigger source

ADTS2	ADTS1	ADTS0	Comment
0	0	0	Free running mode
0	0	1	Analog comparator
0	1	0	External interrupt request 0
0	1	1	Timer/couter0 compare match A
1	0	0	Timer/counter0 overflow
1	0	1	Timer/couter1 compare match B
1	1	0	Timer/counter1 overflow
1	1	1	Timer/counter1 capture event

Example: Free Running Mode w/ Interrupt

- Read data from ADC0 and displays the result on Port B and Port D indefinitely
- Enable ADC and select ADC clock to be ck/128

  ⇒ **ADCSRA** = 0xAF **ADCSRB** = 0x00

	1		1		1	1	1	1
ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
ADCSRB	-	ACME	-	-	-	ADTS2	ADTS1	ADTS0
						0	0	0

- Select 1.1V internal reference voltage and ADC0

  ⇒ **ADMUX** = 0xC0

	1	1			0	0	0	0
ADMUX	REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0

Flowchart (Free Running)

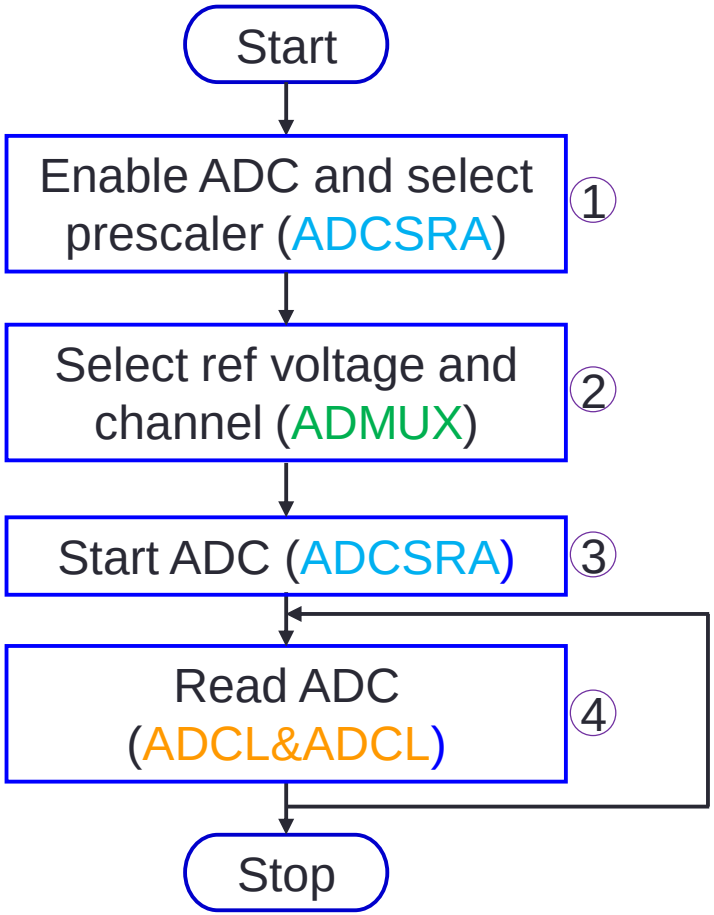
- What value do we set the registers?

ADCSRA

1		1			1	1	1
ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0

REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0
1	1	0		0	0	0	0

ADMUX



Read DMS Sensor (Free Running)

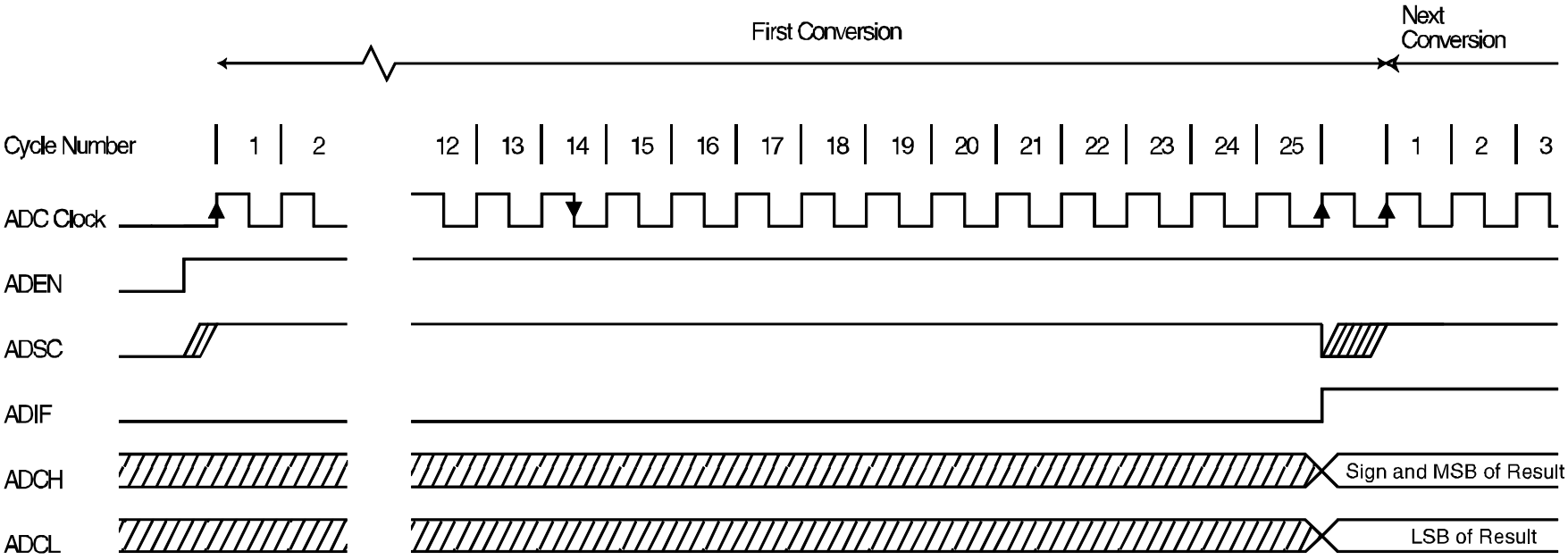
```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
    CLKPR=(1<<CLKPCE);
    CLKPR=0b00000011;                    // set clk to 1Mhz
    DDRB=0xFF;                           // PORTB as output
    DDRD=0xFF;                           // PORTD as output
    DDRC=0;                              // PORTC as input
    ADCSRA=0b10100111;                   // free running mode
    ADMUX=0b11000000;                    // ref volt + channel
    ADCSRA|=(1<<ADSC);                   // start ADC
    while (1) {
        PORTD=ADCL;                      // read low byte first
        PORTB=ADCH;
        _delay_ms(200);
    }
}
```

Note: Do NOT need to check the flag ADIF

ADC Conversion Time

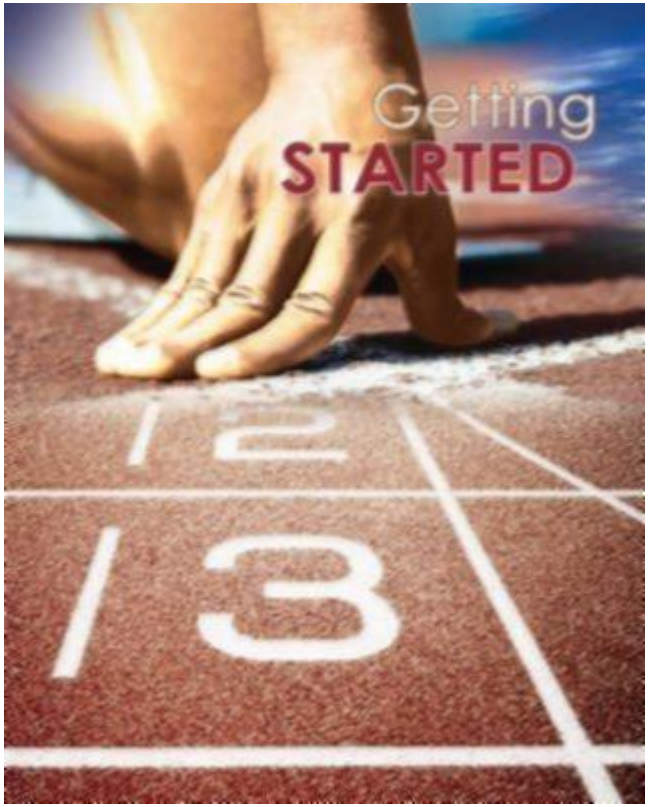
- ADC Timing Diagram, First Conversion



Condition	Sample & Hold (cycle)	Conversion Time (Cycle)
First conversion	13.5	25
Normal conversions, single ended	1.5	13
Auto triggered conversions	2	13.5

Outline (Cont'd)

- Analog-to-digital converter (ADC)
  - Analog and digital signals
  - Successive approximation ADC
  - AVR ADC connection
  - ADC registers
  - Single conversion mode
  - Free running mode
- Getting started



Reference

- ATmega328P data sheet
- AVR 8-bit instruction set
- Atmel AVR126: ADC of megaAVR in Single Ended Mode
- Atmel AVR127: Understanding ADC parameters
- AVR131: Using the AVR's High-speed PWM
- M. A. Mazidi, S. Naimi, and S. Naimi, *The AVR Microcontroller and Embedded Systems: Using Assembly and C*, Prentice Hall, 2010
- AVR GCC library help http://nongnu.org/avr-libc/user-manual/modules.html