1 Features

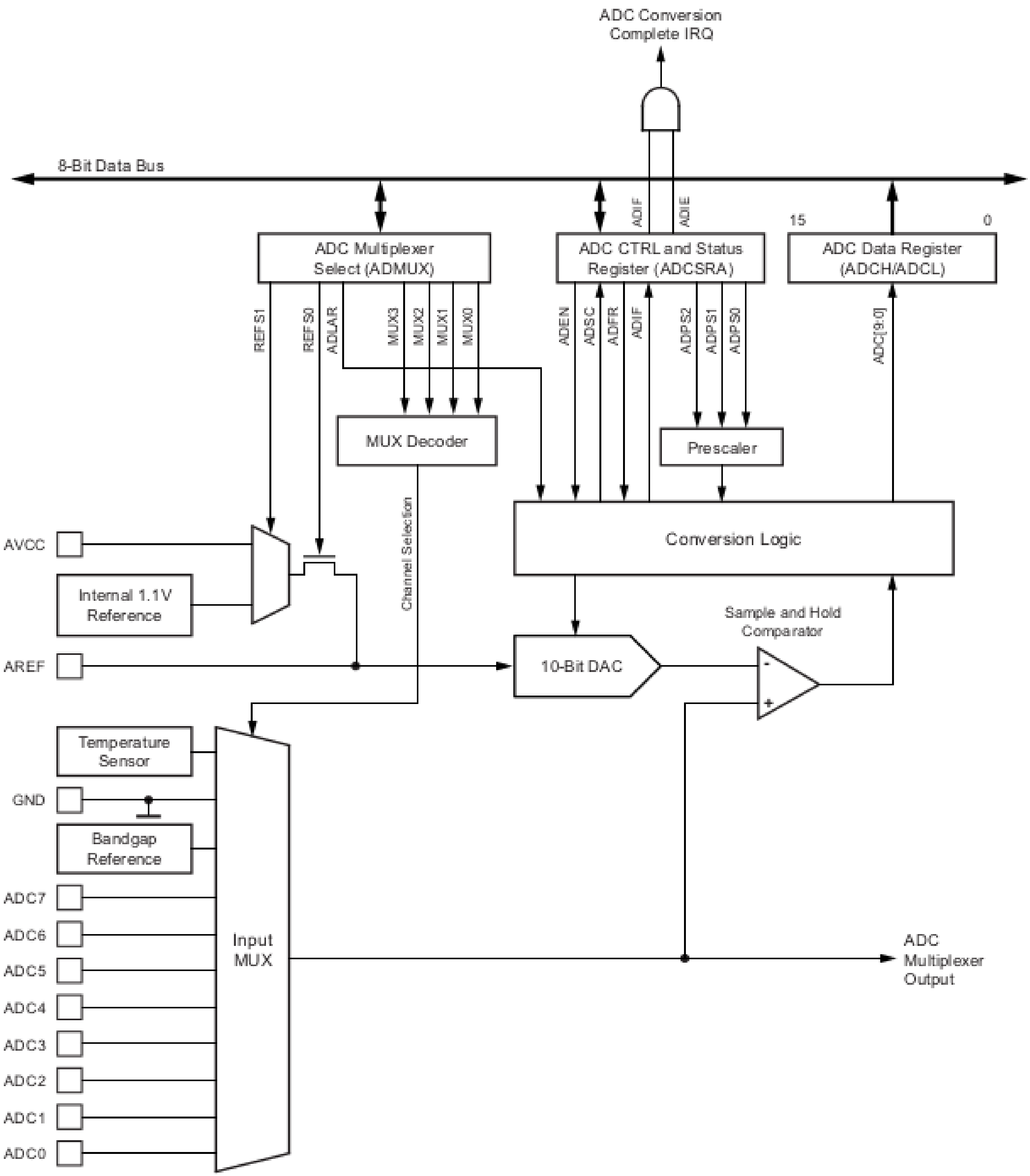
- 10-bit successive approximation ADC
- 65 to 260 μs conversion time
- 15 kilo samples per second
- 6 Multiplexed single ended input channels
- 2 Additional multiplexed single ended input channels depending on the package
- Temperature Sensor input Channel
- Selectable 1.1V ADC reference voltage
- Free running or singe conversion mode
- Interrupt on ADC conversion complete

2 Overview

- Minimum value = 0V and Maximum value = $V_{REF} - 1$ LSB
- AV_{CC} can should be $V_{CC}\pm$ 0.3V.
- The ***MUX*** bits in **ADMUX** register is used to select either ADC input pins or GND or Temperature Sensor or fixed band gap voltage reference(1.1V) for single ended input of ADC.
- The input clock frequency of ADC must be between 50kHz and 200kHz for max. resolution.
- Normal conversion takes 13 ADC clock cycles.
- The adc output are stored in **ADCH** and **ADCL** register.
- Can choose output between left or right adjusted by ***ADLAR*** bit in ADMUX.

Notes : First read **ADCH** and then read **ADCL**.

3 Block Diagram



4 Starting Conversion

4.1 Single Conversion

- Disabling the power reduction ADC bit (***PRADC***).
- Writing logical one to ADC start conversion bit (***ADSC***).
- This Start conversion bit is cleared by hardware when ADC completes conversion.

4.2 Triggered Conversion

- Many sources can be used to trigger.

- Auto trigger is enabled by setting ADC auto trigger enable bit(***ADATE***) in **ADCSRA** register.
- Trigger source is selected by ADC trigger select bits (***ADTS***) in **ADCSRB** register.
- When positve edge occur on selected trigger signal, the ADC starts conversion.
- Until the ADC conversion ends and another positve edge occur on selected trigger source, the next conversion wont's tart.

4.2.1 Free Running Mode

- Using ADC interrupt Flag as trigger source makes the ADC start new conversion as soon as ongoing conversion ends.
- This is the free running mode, when constant sampling and updating is done.
- The first conversion is started by setting the ***ADSC*** bit **ADCSRA** register.
- No need to clear interupt flag.

Note : The ***ADSC*** bit can be used to check if the conversion is going on or not independent of the mode.

5 Register Description

ADMUX – ADC Multiplexer Selection Register

7	6	5	4	3	2	1	0
REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0

- ***ADLAR*** - ADC Left Adjust Result - presentation of ADC conversion results.[1 - Left adjusted; 0 - Right adjusted]

REFS[1:0]	**Voltage Reference**
00	AREF - the actual reference voltage
01	AV_{CC}
10	Reserved
11	Internal 1.1V

MUX[3:0]	**Single Ended Input**
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	Temperature Sensor
1110	1.1V Internal Voltage Reference
111	0V

ADCSRA – ADC Control and Status Register A

7	6	5	4	3	2	1	0
REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0

- ***ADEN*** - ADC Enable - enabled the ADC.
- ***ADSC*** - ADC Start Conversion - starts the conversion in singe conversion mode and start first conversion in free running mode.
- ***ADATE*** - ADC Auto Trigger Enable - auto triggering the ADC on positve edge of selected trigger signal.
- ***ADIF*** - ADC Interrupt Flag - indicates the End of conversion.
- ***ADIE*** - ADC Interrupt Enable- enables the ADC conversion complete interrupt.

ADTS[2:0] - **ADC Auto Trigger Source Selections**	**Trigger Source**
000	Free running mode
001	Analog comparator
010	External interrupt request 0
011	Timer/Counter0 compare match A
100	Timer/Counter0 overflow
101	Timer/Counter1 compare match B
110	Timer/Counter1 overflow
111	Timer/Counter1 capture event

ADPS[2:0] - **ADC Prescaler Select**	**Division Factor**
000	2
001	2
010	4
011	8
100	16
101	32
110	64
111	128

ADCSRB – ADC Control and Status Register B

7	6	5	4	3	2	1	0
–	ACME	-	-	-	ADTS2	ADTS1	ADTS0

ADCL and ADCH – The ADC Data Register

ADLAR=0

15	14	13	12	11	10	9	8
-	-	-	-	-	-	ADC[9:8]	
ADC[7:0]							
7	6	5	4	3	2	1	0

ADLAR=1

15	14	13	12	11	10	9	8
ADC[9:2]							
ADC[1:0]		-	-	-	-	-	-
7	6	5	4	3	2	1	0

6 Configuring the ADC

6.1 Single Conversion

- First, Voltage Reference is choosen by configuring the ***REFS[1:0]*** bits in **ADMUX** register.
- Next, the ADC output presentation either left or right adjusting is choosen by configuring the ***ADLAR*** bit in **ADMUX** register.
- Next, the channel is choosen by configuring the ***MUX[3:0]*** bits in **ADMUX** register.
- Next, for single conversion, the ***ADATE*** - ADC auto trigger bit is cleared in **ADCSRA** register.
- Interrupt is disbaled, as we use single conversion every time in program by clearing the ***ADIE*** bit in **ADCSRA** register.

- The Prescaler for ADC clock is choosen so that the clock is between 50kHz and 200kHz by Configuring the ***ADPS[2:0]*** bits in **ADCSRA** register.
- ADC is enabled by seting the ***ADEN*** bit in **ADCSRA** register.
- Finally, the ADC conversion is started by setting the ***ADSC*** bit in **ADCSRA** register.
- Next, we check the ***ADSC*** flag for end of conversion.
- We can read the output from ***ADC*** register.

```
DDRC &= ~(1<<channel_no);

// Selecting Voltage Referece
// Lets use AREF pin
// REFS[1:0] -- 00
ADMUX &= ~(1<<REFS0);
ADMUX &= ~(1<<REFS1);

// Selecting the Presentation of ADC output
// Right adjust - ADLAR == 0
ADMUX &= ~(1<<ADLAR);

// SELECTINT the channel for ADC
// LET'S select channel_no
// MUX[3:0]&0xF0 | channel_no
ADMUX = (ADMUX & 0XF0) | channel_no;

// for single conversion - disabling ADC auto trigger
// ADATE == 0
ADCSRA &= ~(1<<ADATE);

// disable the interrrupt by disbaling ADIE bit
// ADIE == 0
ADCSRA &= ~(1<<ADIE);

//  Prescaler be 64 so that we get 8Mhz/64 = 125kHz
// ADPS[2:0] -- 110
ADCSRA |= (1<<ADPS2) | (1<<ADPS1);
ADCSRA &= ~(1<<ADPS0);

// ENABLING adc
ADCSRA |= (1<<ADEN);

// STARTING CONVERSIOn
ADCSRA |= (1<<ADSC);

    // since single conversion, we can check start conversion bit
while((ADCSRA & (1<<ADSC)))
{

}
// RESSETTING THE Flag
// ADCSRA |= (1<<ADIF);
return ADC;
```

6.2 Free Running Conversion

- First, Voltage Reference is choosen by configuring the ***REFS[1:0]*** bits in **ADMUX** register.
- Next, the ADC output presentation either left or right adjusting is choosen by configuring the ***ADLAR*** bit in **ADMUX** register.
- Next, the channel is choosen by configuring the ***MUX[3:0]*** bits in **ADMUX** register.

- Next, the trigger source of auto trigger is choosen by selecting 000 (free running) in ***ADTS[2:0]*** bits in **ADCSRA** register.
- Next, for Free Running conversion, the ***ADATE*** - ADC auto trigger bit is set in **ADCSRA** register.
- Interrupt is enabled by setting the ***ADIE*** bit in **ADCSRA** register.
- The Prescaler for ADC clock is choosen so that the clock is between 50kHz and 200kHz by Configuring the ***ADPS[2:0]*** bits in **ADCSRA** register.
- ADC is enabled by seting the ***ADEN*** bit in **ADCSRA** register.
- Finally, the ADC conversion is started by setting the ***ADSC*** bit in **ADCSRA** register.
- Next, we write a ISR for handling the End of conversion.

```
DDRC &= ~(1<<channel_no);

// Selecting Voltage Referece
// Lets use AREF pin
// REFS[1:0] -- 00
ADMUX &= ~(1<<REFS0);
ADMUX &= ~(1<<REFS1);

// Selecting the Presentation of ADC output
// Right adjust - ADLAR == 0
ADMUX &= ~(1<<ADLAR);

// SELECTINT the channel for ADC
// LET'S select channel_no
// MUX[3:0]&0xF0 | channel_no
ADMUX = (ADMUX & 0XF0) | channel_no;

// Select the Auto Trigger source
// for free running, use 000 for ADTS[2:0] in ADCSRB
ADCSRB &= ~(1<<ADTS2);
ADCSRB &= ~(1<<ADTS1);
ADCSRB &= ~(1<<ADTS0);

// for free runing conversion - enable ADC auto trigger
// ADATE == 1
ADCSRA |= (1<<ADATE);

// enable the interrrupt by enabling ADIE bit
// ADIE == 1
ADCSRA |= (1<<ADIE);

//  Prescaler be 64 so that we get 8Mhz/64 = 125kHz
// ADPS[2:0] -- 110
ADCSRA |= (1<<ADPS2) | (1<<ADPS1);
ADCSRA &= ~(1<<ADPS0);

// ENABLING adc
ADCSRA |= (1<<ADEN);

// STARTING CONVERSIOn
ADCSRA |= (1<<ADSC);

sei();


ISR(ADC_vect)
{
        free_running_value = ADC;
        // ADCSRA |= (1<<ADIF);
}
```