LAMPIRAN A

Listing Program

LISTING PROGRAM BORLAND DELPHI 7.0

Inisialisasi

```
=====================================================
unit ComMainForm;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, CPort, CPortCtl, TeeProcs, TeEngine, Chart, Buttons,
  Series;

type
  TForm1 = class(TForm)
    ComPort: TComPort;
    Memo: TMemo;
    Button_Open: TButton;
    Button_Settings: TButton;
    Panel1: TPanel;
    ComLed1: TComLed;
    ComLed2: TComLed;
    ComLed3: TComLed;
    ComLed4: TComLed;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    ComLed5: TComLed;
    ComLed6: TComLed;
```

```
    Label1: TLabel;
    Label6: TLabel;
    Chart1: TChart;
    BitBtn1: TBitBtn;
    Series1: TLineSeries;
    BitBtn2: TBitBtn;
    Label7: TLabel;
    Label8: TLabel;
    Timer1: TTimer;
    procedure Button_OpenClick(Sender: TObject);
    procedure Button_SettingsClick(Sender: TObject);
    procedure ComPortOpen(Sender: TObject);
    procedure ComPortClose(Sender: TObject);
    procedure ComPortRxChar(Sender: TObject; Count: Integer);
    procedure Bt_LoadClick(Sender: TObject);
    procedure Bt_StoreClick(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure MemoChange(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

Menu

```
=======================================================
var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.Button_OpenClick(Sender: TObject);
begin
  if ComPort.Connected then
    ComPort.Close

  else
    ComPort.Open;
end;

procedure TForm1.Button_SettingsClick(Sender: TObject);
begin
  ComPort.ShowSetupDialog;
end;

procedure TForm1.ComPortOpen(Sender: TObject);
begin
  Button_Open.Caption := 'Close';
end;
```

```
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
Application.Terminate;
end;

procedure TForm1.BitBtn2Click(Sender: TObject);
begin
series1.Clear;
end;

procedure TForm1.ComPortClose(Sender: TObject);
begin
  if Button_Open <> nil then
    Button_Open.Caption := 'Open';
end;
```

Terima data serial

```
==========================================================
procedure TForm1.ComPortRxChar(Sender: TObject; Count: Integer);
var
  Str: String;
begin
  ComPort.ReadStr(Str, Count);
  Memo.Text := Memo.Text + Str;
  with chart1 do
  with series1 do
  addy(strtofloat(Memo.Text + Str)/10,'',clblue);
  end;
```

```
procedure TForm1.MemoChange(Sender: TObject);
begin
memo.Clear;
end;
```

Jam

```
==========================================================
procedure TForm1.Timer1Timer(Sender: TObject);
begin
label8.Caption:=timetostr(time);
end;

end.
```

LISTING PROGRAM CODE VISION C COMPILER

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.25.3 Professional
Automatic Program Generator
© Copyright 1998-2007 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : SENSOR ANGIN
Version :
Date    : 7/23/2008
Author  : F4CG
Company : F4CG
Comments:

Chip type           : ATmega16
Program type        : Application
Clock frequency     : 11.059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************************/

#include <mega16.h>
#include <stdlib.h>
#include <delay.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x18 ;PORTB
#endasm
#include <lcd.h>
```

```
// Standard Input/Output functions
#include <stdio.h>
unsigned int a=0;

unsigned char ca[10];
// Timer 1 overflow interrupt service routine
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Reinitialize Timer 1 value
TCNT1H=0xD5;
TCNT1L=0xD0;
// Place your code here

printf("%d",a);
lcd_gotoxy(5,0);
itoa(a,ca);
lcd_puts(ca);
a=0;

}

// Declare your global variables here

void tunggu(void)
{
bit sekarang;
sekarang=PINA.0;
```

```
cek:
if(PINA.0==sekarang)
goto cek;
return;

}

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out
Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;
```

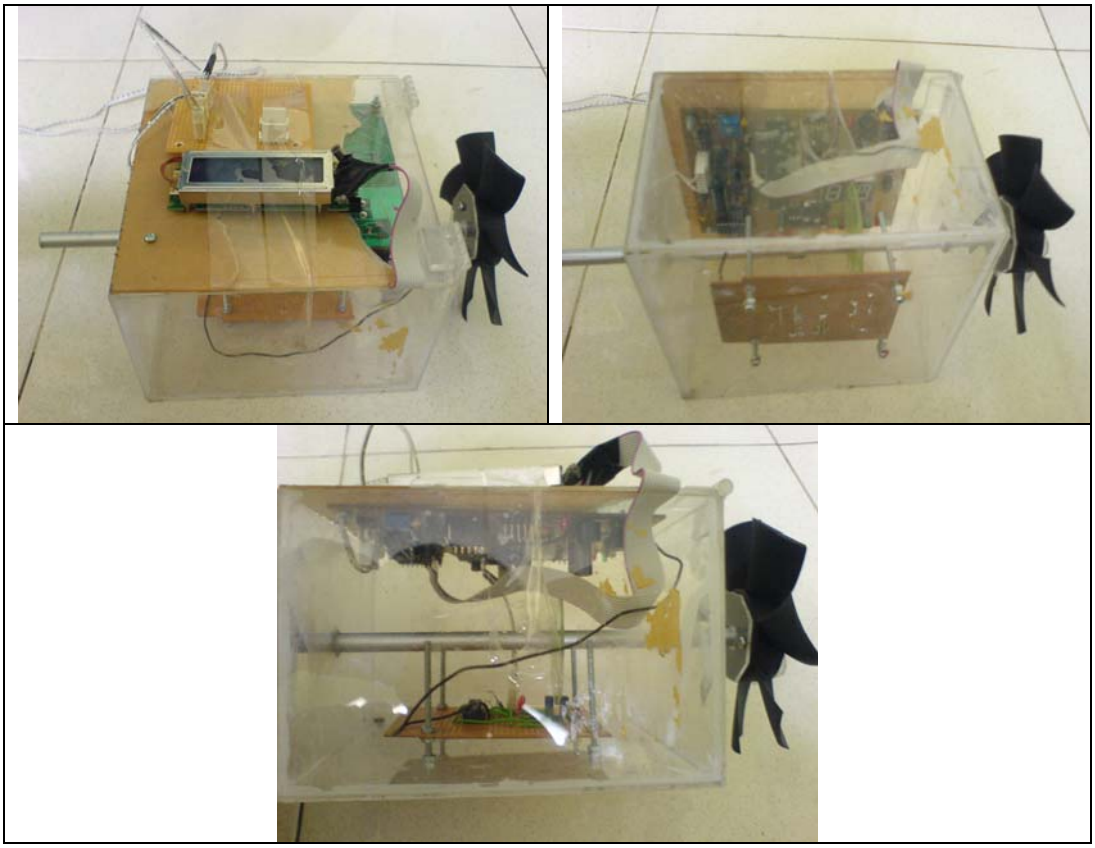
```
// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 10.800 kHz
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x05;
TCNT1H=0xD5;
TCNT1L=0xD0;
```

```
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x04;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
```

```
// USART Baud rate: 9600
UCSRA=0x00;
UCSRB=0x18;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// LCD module initialization
lcd_init(16);
lcd_gotoxy(0,1);
lcd_putsf(" = Wind Speed =");
lcd_gotoxy(11,0);
lcd_putsf("m/s");
// Global enable interrupts
#asm("sei")
while (1)
      {
      // Place your code here

if (PINA.0==1)
{a++;tunggu();
 }


      };
}
```

LAMPIRAN B

Prototipe



Kalibrasi Alat

Kalibrasi Alat Ukur

A (m/s)	B (Jumlah Pulsa)	C (Sensor, m/s)	D (A/C)
20.6	206	4.45	4.63
20	200	4.32	4.63
18.1	181	3.91	4.64
17.7	177	3.82	4.63
16.6	165	3.57	4.64
14.4	143	3.09	4.63
12.4	124	2.68	4.63
10.7	107	2.31	4.63
9.4	94	2.03	4.63
8.8	88	1.9	4.63
	RATA-RATA D		4.63

Keterangan :
A=Kecepatan Alat Ukur Standar (Lab Tekanan Direktorat Metrologi, Pasteur)
B=Banyaknya Pulsa (Pada Sensor)
C=Kecepatan Sensor
D=Faktor Kesalahan

PERCOBAAN

LAMPIRAN C

IC4093

INTEGRATED CIRCUITS

DATA SHEET

For a complete data sheet, please also download:

- The IC04 LOCMOS HE4000B Logic Family Specifications HEF, HEC
- The IC04 LOCMOS HE4000B Logic Package Outlines/Information HEF, HEC

HEF4093B
gates
Quadruple 2-input NAND Schmitt trigger

Product specification
File under Integrated Circuits, IC04

January 1995

Philips Semiconductors

PHILIPS

DESCRIPTION

The HEF4093B consists of four Schmitt-trigger circuits. Each circuit functions as a two-input NAND gate with Schmitt-trigger action on both inputs. The gate switches at different points for positive and negative-going signals. The difference between the positive voltage (V_P) and the negative voltage (V_N) is defined as hysteresis voltage (V_H).

Fig.2 Pinning diagram.

HEF4093BP(N): 14-lead DIL; plastic (SOT27-1)

HEF4093BD(F): 14-lead DIL; ceramic (cerdip) (SOT73)

HEF4093BT(D): 14-lead SO; plastic (SOT108-1)

( ): Package Designator North America

Fig.3 Logic diagram (one gate).

FAMILY DATA, I_{DD} LIMITS category GATES

See Family Specifications

Fig.1 Functional diagram.

DC CHARACTERISTICS

V_{SS} = 0 V; T_{amb} = 25 °C

	V_{DD} V	SYMBOL	MIN.	TYP.	MAX.	
Hysteresis voltage	5	V_H	0.4	0.7	–	V
	10		0.6	1.0	–	V
	15		0.7	1.3	–	V
Switching levels positive-going input voltage	5	V_P	1.9	2.9	3.5	V
	10		3.6	5.2	7	V
	15		4.7	7.3	11	V
negative-going input voltage	5	V_N	1.5	2.2	3.1	V
	10		3	4.2	6.4	V
	15		4	6.0	10.3	V

Fig.4 Transfer characteristic.

Fig.5 Waveforms showing definition of V_P, V_N and V_H; where V_N and V_P are between limits of 30% and 70%.

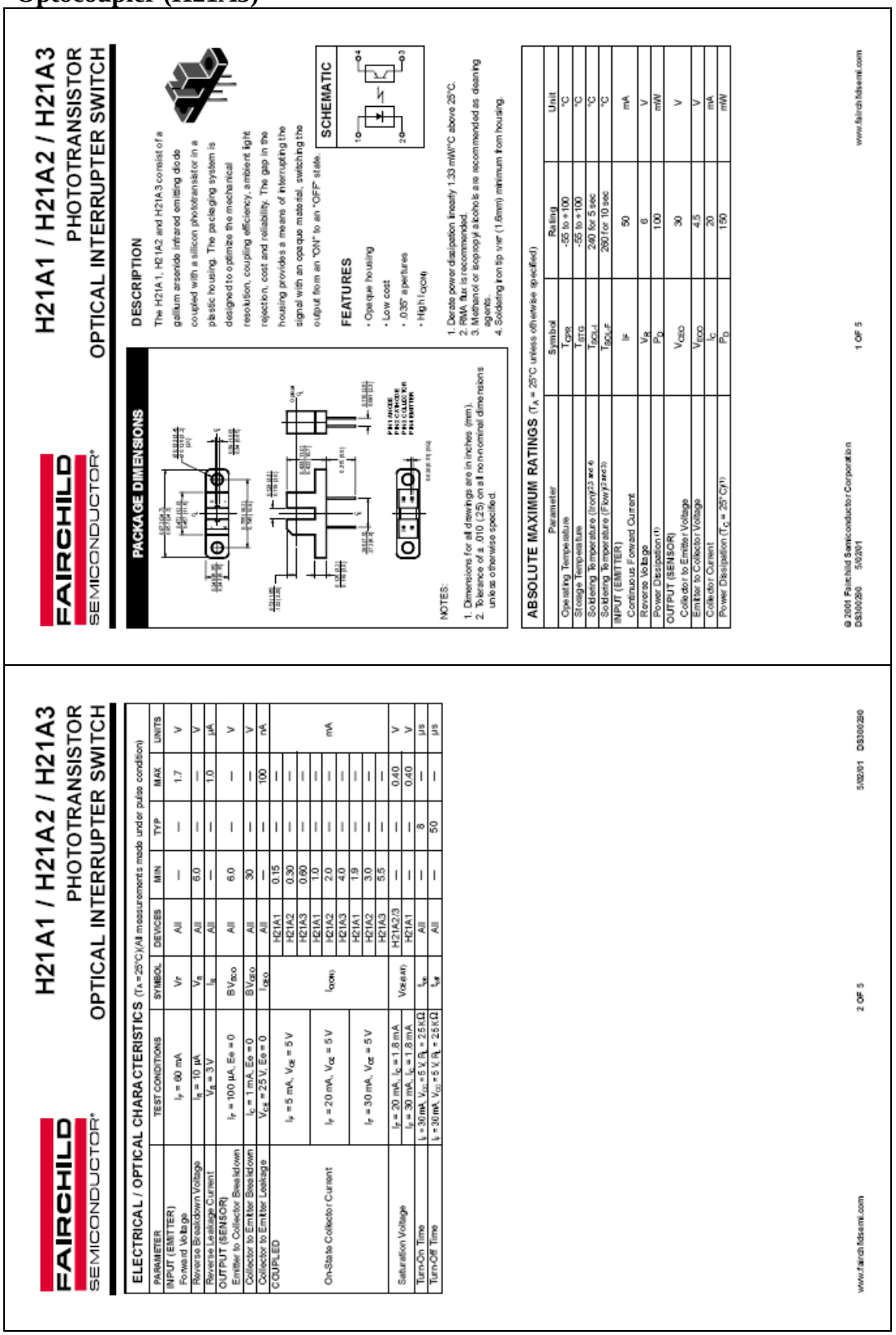
AC CHARACTERISTICS

V_{SS} = 0 V; T_{amb} = 25 °C; C_L = 50 pF; input transition times ≤ 20 ns

	V_{DD} V	SYMBOL	TYP.	MAX.	TYPICAL EXTRAPOLATION FORMULA
Propagation delays $I_n \rightarrow O_n$ HIGH to LOW	5	t_{PHL}	80 ns	185 ns	63 ns + (0.55 ns/pF) C_L
	10		40 ns	80 ns	29 ns + (0.23 ns/pF) C_L
	15		30 ns	60 ns	22 ns + (0.16 ns/pF) C_L
LOW to HIGH	5	t_{PLH}	85 ns	170 ns	58 ns + (0.55 ns/pF) C_L
	10		40 ns	80 ns	29 ns + (0.23 ns/pF) C_L
	15		30 ns	60 ns	22 ns + (0.16 ns/pF) C_L
Output transition times HIGH to LOW	5	t_{THL}	60 ns	120 ns	10 ns + (1.0 ns/pF) C_L
	10		30 ns	60 ns	9 ns + (0.42 ns/pF) C_L
	15		20 ns	40 ns	6 ns + (0.28 ns/pF) C_L
LOW to HIGH	5	t_{TLH}	60 ns	120 ns	10 ns + (1.0 ns/pF) C_L
	10		30 ns	60 ns	9 ns + (0.42 ns/pF) C_L
	15		20 ns	40 ns	6 ns + (0.28 ns/pF) C_L

	V_{DD} V	TYPICAL FORMULA FOR P (µW)	
Dynamic power dissipation per package (P)	5	1300 f_i + $\sum(f_o C_L) \times V_{DD}^2$	where f_i = input freq. (MHz) f_o = output freq. (MHz) C_L = load capacitance (pF) $\sum(f_o C_L)$ = sum of outputs V_{DD} = supply voltage (V)
	10	6400 f_i + $\sum(f_o C_L) \times V_{DD}^2$	
	15	18 700 f_i + $\sum(f_o C_L) \times V_{DD}^2$	

Optocoupler (H21A3)



H21A1 / H21A2 / H21A3
PHOTOTRANSISTOR
OPTICAL INTERRUPTER SWITCH

FAIRCHILD
SEMICONDUCTOR®

ELECTRICAL / OPTICAL CHARACTERISTICS (T_A=25°C)(All measurements made under pulse condition)

PARAMETER	TEST CONDITIONS	SYMBOL	DEVICES	MIN	TYP	MAX	UNITS
INPUT (EMITTER) Forward Voltage	I_F = 60 mA	V_F	All	—	—	1.7	V
Reverse Breakdown Voltage	I_R = 10 µA	V_R	All	6.0	—	—	V
Reverse Leakage Current	V_R = 3 V	I_R	All	—	—	1.0	µA
OUTPUT (SENSOR) Emitter to Collector Breakdown	I_F = 100 µA, Ee = 0	BV$_{ECO}$	All	6.0	—	—	V
Collector to Emitter Breakdown	I_C = 1 mA, Ee = 0	BV$_{CEO}$	All	30	—	—	V
Collector to Emitter Leakage	V_{CE} = 25 V, Ee = 0	I_{CEO}	All	—	—	100	nA
COUPLED On-State Collector Current	I_F = 5 mA, V_{CE} = 5 V	$I_{C(ON)}$	H21A1	0.15	—	—	mA
			H21A2	0.30	—	—	
			H21A3	0.60	—	—	
	I_F = 20 mA, V_{CE} = 5 V		H21A1	1.0	—	—	
			H21A2	2.0	—	—	
			H21A3	4.0	—	—	
	I_F = 30 mA, V_{CE} = 5 V		H21A1	1.9	—	—	
			H21A2	3.0	—	—	
			H21A3	5.5	—	—	
Saturation Voltage	I_F = 20 mA, I_C = 1.8 mA	V$_{CE(SAT)}$	H21A2/3	—	—	0.40	V
	I_F = 30 mA, I_C = 1.8 mA		H21A1	—	—	0.40	V
Turn-On Time	I_C = 30 mA, V_{CC} = 5 V, R_L = 25 kΩ	t_{on}	All	—	8	—	µs
Turn-Off Time	I_C = 30 mA, V_{CC} = 5 V, R_L = 25 kΩ	t_{off}	All	—	50	—	µs

H21A1 / H21A2 / H21A3
PHOTOTRANSISTOR
OPTICAL INTERRUPTER SWITCH

FAIRCHILD
SEMICONDUCTOR®

PACKAGE DIMENSIONS

NOTES:
1. Dimensions for all drawings are in inches (mm).
2. Tolerance of ± .010 (.25) on all non-nominal dimensions unless otherwise specified.

DESCRIPTION

The H21A1, H21A2 and H21A3 consist of a gallium arsenide infrared emitting diode coupled with a silicon phototransistor in a plastic housing. The packaging system is designed to optimize the mechanical resolution, coupling efficiency, ambient light rejection, cost and reliability. The gap in the housing provides a means of interrupting the signal with an opaque material, switching the output from an "ON" to an "OFF" state.

FEATURES

- Opaque housing
- Low cost
- .035" apertures
- High I_C/I_F(ON)

SCHEMATIC

1. Derate power dissipation linearly 1.33 mW/°C above 25°C.
2. RMA flux is recommended.
3. Methanol or isopropyl alcohols are recommended as cleaning agents.
4. Soldering iron tip 1/16" (1.6mm) minimum from housing.

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C unless otherwise specified)

Parameter	Symbol	Rating	Unit
Operating Temperature	T_{OPR}	-55 to +100	°C
Storage Temperature	T_{STG}	-55 to +100	°C
Soldering Temperature (Iron)(3 sec.)	T_{SOL-I}	240 for 5 sec	°C
Soldering Temperature (Flow)(2 sec.)	T_{SOL-F}	260 for 10 sec	°C
INPUT (EMITTER) Continuous Forward Current	I_F	50	mA
Reverse Voltage	V_R	6	V
Power Dissipation (1)	P_D	100	mW
OUTPUT (SENSOR) Collector to Emitter Voltage	V_{CEO}	30	V
Emitter to Collector Voltage	V_{ECO}	4.5	V
Collector Current	I_C	20	mA
Power Dissipation (T_C = 25°C)(1)	P_D	150	mW

LCD M1632 Seiko

SII
Seiko Instruments

LCD MODULES

WITH BUILT-IN LED BACKLIGHT

TECHNICAL DATA

TD.No.LED-071E

1. GENERAL

1.1 General

The LCD modules with built-in LED backlight consist of an LCD module and an LED backlight. The LCD is lit from behind by the LED, so it can be read clearly and easily, even in dark places.

Since the transflective-type reflector is used, a choice of ambient lighting during daylight or LED backlighting at night is possible. By switching off the LED in bright places, the current consumption can be reduced.

Seiko Instruments inc. offers two versions of LCD modules with built-in LED backlight : the TN-LCD module and the New-TN-LCD module. The New-TN LCD features wider viewing angle and higher contrast than TN-LCD.

Features of LED backlight

- Brightness
- Low voltage
- Long service life : 50,000 hours (average life)
- Color of light : Yellow-green
- Peak light-emitting wavelength : 570 nm

As for the operating instructions and interfacing, refer to our *Liquid Crystal Display Module User Manual* issued for each module (M1641, M1632 etc.)

- 1 -

1.3 Block Diagram

Figure 1

1.4 Absolute Maximum Ratings

V_{SS} = 0 V

Item	Symbol	Standard	Unit	Remarks
Power supply voltage	V_{DD}	– 0.3 to + 7.0	V	
	V_{LC}	V_{DD} – 13.5 to V_{DD} + 0.3	V	
Input voltage	V_{in}	– 0.3 to V_{DD} + 0.3	V	
Operating temperature	T_{opr}	0 to + 50	°C	
Storage temperature	T_{stg}	– 20 to + 60	°C	At 50% RH

1.5 Electrical Characteristics

V_{DD} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0°C to 50°C

Item		Symbol	Condition	Min.	Typ.	Max.	Unit
Input voltage*	High	V_{IH1}		2.2	–	V_{DD}	V
	Low	V_{IL1}		0	–	0.6	V
Output voltage**	High	V_{OH1}	$-I_{OH}$ = 0.205 mA	2.4	–	–	V
	Low	V_{OL1}	I_{OL} = 1.2 mA	–	–	0.4	V
Power supply voltage		V_{DD}		4.75	5.00	5.25	V
		V_{DD}-V_{LC}		1.5	–	–	V
Current consumption		I_{DD}		–	2.0	3.0	mA
		I_{LC}	V_{LC} = 0.25V	–	0.2	1.0	mA
Clock oscillation freq.		f_{osc}	Resistance oscillation	190	270	350	kHz

* Applied to DB_0 to DB_7, E, R/$\overline{W}$, and RS
** Applied to DB_0 to DB_7

Remark: Recommended operating voltage

The viewing angle and screen contrast of the LCD panel can be varied by changing the liquid crystal operating voltage (V_{OPR}), that is V_{LC}.
The optical characteristics is influenced by an ambient temperature. The recommended value of V_{OPR} for an ambient temperatures are shown below.

Temperature (°C)	0	10	25	40	50
V_{OPR} (V)	5.00	4.90	4.75	4.60	4.50

$V_{OPR} = V_{DD} - V_{LC}$

AN.No.1632-842E

1.6 Timing Characteristics

1.6.1 Write operation

V_{DD} = 5.0 V ± 5%, V_{SS} = 0 V, Ta = 0°C to 50°C

Item		Symbol	Min.	Max.	Unit
Enable cycle time		t_{CycE}	1000	–	ns
Enable pulse width	High level	PW_{EH}	450	–	ns
Enable rise and fall time		t_{Er},t_{Ef}	–	25	ns
Setup time	RS, R/$\overline{W}$ → E	t_{AS}	140	–	ns
Address hold time		t_{AH}	10	–	ns
Data setup time		t_{DSW}	195	–	ns
Data hold time		t_H	10	–	ns

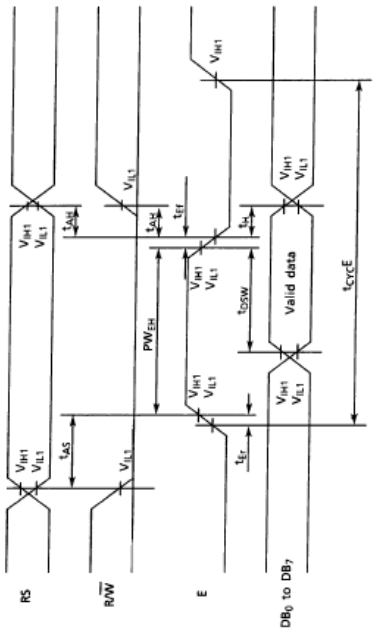


Figure 2 Data write from MPU to module

AN.No.1632-842E

1.6.2 Read operation

V_{DD} = 5.0 V ± 5%, V_{SS} = 0 V, Ta = 0°C to 50°C

Item		Symbol	Min.	Max.	Unit
Enable cycle time		t_{CycE}	1000	–	ns
Enable pulse width	High level	PW_{EH}	450	–	ns
Enable rise and fall time		t_{Er},t_{Ef}	–	25	ns
Setup time	RS, R/$\overline{W}$→E	t_{AS}	140	–	ns
Address hold time		t_{AH}	10	–	ns
Data delay time		t_{DDR}	–	320	ns
Data hold time		t_H	20	–	ns

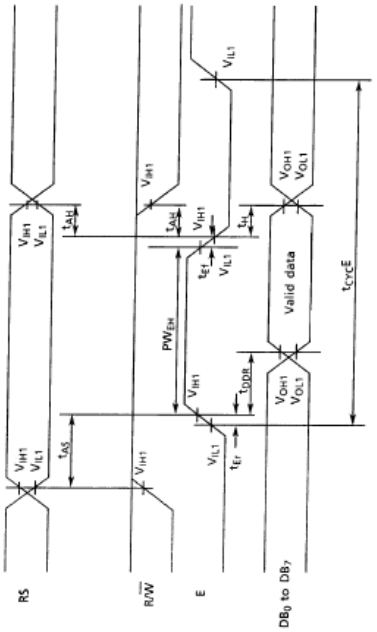


Figure 3 Data read from module to MPU

AN.No.1632-842E

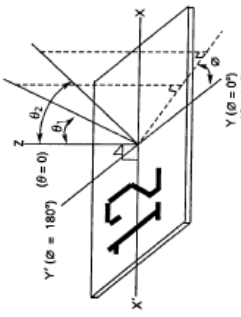
1.7 Optical Characteristics

1.7.1 Optical characteristics (TN LCD)

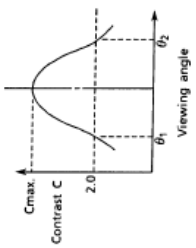
Viewing angle: 6 o'clock (ϕ = 0°)
Ta = 25°C, V_{opr} = 4.75 V

Item	Symbol	Conditions	Min.	Typ.	Max.	Remarks
Viewing angle	$\theta_2 - \theta_1$	$C \geq 2.0$, ϕ = 0°	35	–	–	See Notes 1 and 2
Contrast	C	θ = 25° , ϕ = 0°	5	8	–	See Note 3
Response time (rise)	t_{on}	θ = 25° , ϕ = 0°	–	60 ms	70 ms	See Note 4
Response time (fall)	t_{off}	θ = 25° , ϕ = 0°	–	150 ms	170 ms	See Note 4

Note 1: Definition of angles ϕ and θ

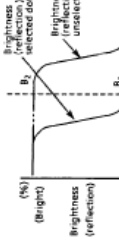


Note 2: Definition of viewing angles θ_1 and θ_2

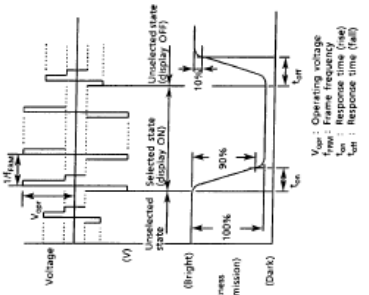


Note 3: Definition of contrast C

$C = \frac{\text{Brightness (reflection) of unselected dot (B2)}}{\text{Brightness (reflection) of selected dot (B1)}}$

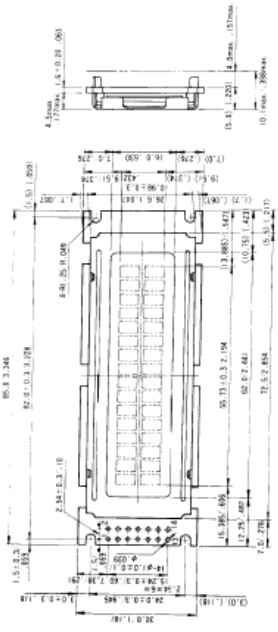


Note 4: Definition of response time



AN.No.1632-842E

1.8 Dimensions



Unit : mm/inch
General tolerance : ± 0.5 mm

I/O terminal symbol

No.	Symbol
1	DB_7
2	DB_6
3	DB_5
4	DB_4
5	DB_3
6	DB_2
7	DB_1
8	DB_0
9	E
10	R/$\overline{W}$
11	RS
12	V_{LC}
13	V_{SS}
14	V_{DD}

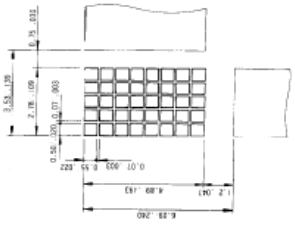


Figure 4

2. OPERATING INSTRUCTIONS

2.1 Terminal Functions

Table 1 Terminal functions

Signal name	No. of terminals	I/O	Destination	Function
DB_0 to DB_3	4	I/O	MPU	Tristate bidirectional lower four data buses: Data is read from the module to the MPU or written to the module from the MPU through the buses. If the interface data is 4 bits, the signals are not used.
DB_4 to DB_7	4	I/O	MPU	Tristate bidirectional upper four data buses: Data is read from the module to the MPU or written to the module from the MPU through the buses. DB_7 is also used as a busy flag.
E	1	Input	MPU	Operation start signal: The signal activates data write or read.
R/$\overline{W}$	1	Input	MPU	Read (R) and Write ($\overline{W}$) selection signals 0 : Write 1 : Read
RS	1	Input	MPU	Register selection signals 0 : Instruction register (Write) Busy flag and address counter (Read) 1 : Data register (Write and Read)
V_{LC}	1	–	Power supply	Power supply terminal for driving liquid crystal display: The screen contrast can be varied by changing V_{LC}.
V_{DD}	1	–	Power supply	+5 V
V_{SS}	1	–	Power supply	Ground terminal: 0 V

2.2 Basic Operations

2.2.1 Registers

The controller has two kinds of eight-bit registers: the instruction register (IR) and the data register (DR). They are selected by the register select (RS) signal as shown in Table 2.

The IR stores instruction codes such as Display Clear and Cursor Shift, and the address information of display data RAM (DD RAM) and character generator RAM (CG RAM). They can be written from the MPU, but cannot be read to the MPU.

The DR temporarily stores data to be written into DD RAM or CG RAM, or read from DD RAM or CG RAM. When data is written into DD RAM or CG RAM from the MPU, the data in the DR is automatically written into DD RAM or CG RAM by internal operation. However, when data is read from DD RAM or CG RAM, the necessary data address is written into the IR. The specified data is read out to the DR and then the MPU reads it from the DR. After the read operation, the next address is set and DD RAM or CG RAM data at the address is read into the DR for the next read operation.

Table 2 Register selection

RS	R/$\overline{W}$	Operation
0	0	IR selection, IR write. Internal operation : Display clear
0	1	Busy flag (DB_7) and address counter (DB_0 to DB_6) read
1	0	DR selection, DR write. Internal operation : DR to DD RAM or CG RAM
1	1	DR selection, DR read. Internal operation : DD RAM or CG RAM to DR

2.2.2 Busy flag (BF)

The flag indicates whether the module is ready to accept the next instruction. As shown in Table 2, the signal is output to DB_7 if RS = 0 and R/$\overline{W}$ = 1. If the value is 1, the module is working internally and the instruction cannot be accepted. If the value is 0, the next instruction can be written. Therefore, the flag status needs to be checked before executing an instruction. If an instruction is executed without checking the flag status, wait for more than the execution time shown by 2.4 Instruction Outline.

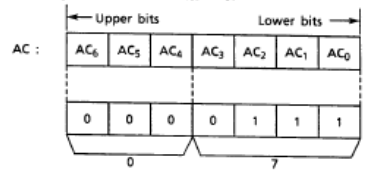
2.2.3 Address counter (AC)

The counter specifies an address when data is written into DD RAM or CG RAM and the data stored in DD RAM or CG RAM is read out. If an Address Set instruction (for DD RAM or CG RAM) is written in the IR, the address information is transferred from the IR to the AC. When display data is written into or read from DD RAM or CG RAM, the AC is automatically incremented or decremented by one according to the Entry Mode Set. The contents of the AC are output to DB_0 to DB_6 as shown in Table 2 if RS = 0 and $R/\overline{W}$ = 1.

2.2.4 Display data RAM (DD RAM)

DD RAM has a capacity of up to 80 × 8 bits and stores display data of 80 eight-bit character codes. Some storage areas of DD RAM which are not used for display can be used as general data RAM.
A DD RAM address to be set in the AC is expressed in hexadecimal form as follows.

Example: DD RAM address = 07



DD RAM addresses 00_H to $0F_H$ are set in the line 1, and 40_H to $4F_H$ in the line 2.

Note : The addresses in the digit 16 of line 1 and the digit 1 of line 2 are not consecutive.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Display digit
Line 1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	DD RAM address
Line 2	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	

If the display is shifted, DD RAM addresses 00_H to 27_H are displayed in line 1 and 40_H to 67_H in line 2. The following figures are examples of display shifts.

*Left shift

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Display digit
Line 1	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	DD RAM address
Line 2	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	

*Right shift

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Display digit
Line 1	27	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	DD RAM address
Line 2	67	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	

2.2.5 Character generator ROM (CG ROM)

Character generator ROM generates 192 types of 5 x 7 dot-matrix character patterns from eight-bit character codes.

Table 3 shows the correspondence between the CG ROM character codes and character patterns.

2.2.6 Character generator RAM (CG RAM)

CG RAM is used to create character patterns freely by programming. Eight types of character patterns can be written.

Table 4 shows the character patterns created from CG RAM addresses and data. To display a created character pattern, the character code in the left column of the table is written into DD RAM corresponding to the display position (digit). The areas not used for display are available as general data RAM.

AN.No.1632-842E

Table 3 Correspondence between character codes and character patterns

Upper 4 bits / Lower 4 bits	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
×××× 0000	CG RAM (1)												
×××× 0001	(2)												
×××× 0010	(3)												
×××× 0011	(4)												
×××× 0100	(5)												
×××× 0101	(6)												
×××× 0110	(7)												
×××× 0111	(8)												
×××× 1000	(1)												
×××× 1001	(2)												
×××× 1010	(3)												
×××× 1011	(4)												
×××× 1100	(5)												
×××× 1101	(6)												
×××× 1110	(7)												
×××× 1111	(8)												

AN.No.1632-842E

Table 4 Relationships between CG RAM addresses and character codes (DD RAM) and character patterns (CG RAM data)

Character code (DD RAM data) 7 6 5 4 3 2 1 0 (Upper bit / Lower bit)	CG RAM address 5 4 3 2 1 0 (Upper bit / Lower bit)	Character pattern (CG RAM data) 7 6 5 4 3 2 1 0 (Upper bit / Lower bit)	
0 0 0 0 * 0 0 0	0 0 0	* * *	Example of character pattern (R)
	0 0 1	* * *	
	0 1 0	* * *	
	0 1 1	* * *	
	1 0 0	* * *	
	1 0 1	* * *	
	1 1 0	* * *	
	1 1 1	* * *	← Cursor position
0 0 0 0 * 0 0 1	0 0 0	* * *	
	0 0 1	* * *	
	0 1 0	* * *	
	0 1 1	* * *	
	1 0 0	* * *	
	1 0 1	* * *	Example of character pattern (↑)
	1 1 0	* * *	
	1 1 1	* * *	
0 0 0 0 * 1 1 1	1 1 1	1 0 0	
		1 0 1	
		1 1 0	
		1 1 1	

Notes: · In CG RAM data, 1 corresponds to Selection and 0 to Non-selection on the display.

· Character code bits 0 to 2 and CG RAM address bits 3 to 5 correspond with each other (three bits, eight types).

· CG RAM address bits 0 to 2 specify a line position for a character pattern. Line 8 of a character pattern is the cursor position where the logical sum of the cursor and CG RAM data is displayed. Set the data of line 8 to 0 to display the cursor. If the data is changed to 1, one bit lights, regardless of the cursor.

· The character pattern column positions correspond to CG RAM data bits 0 to 4 and bit 4 comes to the left end. CG RAM data bits 5 to 7 are not displayed but can be used as general data RAM.

· When reading a character pattern from CG RAM, set to 0 all of character code bits 4 to 7. Bits 0 to 2 determine which pattern will be read out. Since bit 3 is not valid, 00H and 08H select the same character.

AN-No.1632-842E

2.3 Instruction Outline

Table 5 List of Instructions

Instruction	Code RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Function	Execution time*
(1) Display clear	0	0	0	0	0	0	0	0	0	1	Clears all display and returns cursor to home position (address 0)	1.64 ms
(2) Cursor Home	0	0	0	0	0	0	0	0	1	*	Returns cursor to home position. Shifted display returns to home position and DD RAM contents do not change.	1.64 ms
(3) Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets direction of cursor movement and whether display will be shifted when data is written or read	40 µs
(4) Display ON/OFF control	0	0	0	0	0	0	1	D	C	B	Turns ON/OFF total display (D) and cursor (C), and makes cursor position column start blinking (B)	40 µs
(5) Cursor/Display Shift	0	0	0	0	0	1	S/C	R/L	*	*	Moves cursor and shifts display without changing DD RAM contents	40 µs
(6) Function Set	0	0	0	0	1	DL	N	F	*	*	Sets interface data length (DL), the number of display lines (N), and character fonts (F)	40 µs
(7) CG RAM Address Set	0	0	0	1	A_{CG}						Sets CG RAM address to start transmitting or receiving CG RAM data	40 µs
(8) DD RAM Address Set	0	0	1	A_{DD}							Sets DD RAM address to start transmitting or receiving DD RAM data	40 µs
(9) BF/Address Read	0	1	BF	AC							Reads BF indicating module in internal operation and AC contents (Used for both CG RAM and DD RAM)	0 µs
(10) Data Write to CG RAM or DD RAM	1	0	Write Data								Writes data into DD RAM or CG RAM	40 µs
(11) Data Read from CG RAM or DD RAM	1	1	Read Data								Reads data from DD RAM or CG RAM	40 µs

* : Invalid bit
A_{CG} : CG RAM address
A_{DD} : DD RAM address
AC : Address counter

I/D = 1 : Increment
I/D = 0 : Decrement

S = 1 : Display shift
S = 0 : No display shift

D = 1 : Display ON
D = 0 : Display OFF

C = 1 : Cursor ON
C = 0 : Cursor OFF

B = 1 : Blink ON
B = 0 : Blink Off

S/C = 1 : Display shift
S/C = 0 : Cursor movement

R/L = 1 : Right shift
R/L = 0 : Left shift

DL = 1 : 8 bits
DL = 0 : 4 bits

N = 1 : 2 lines
N = 0 : 1 line

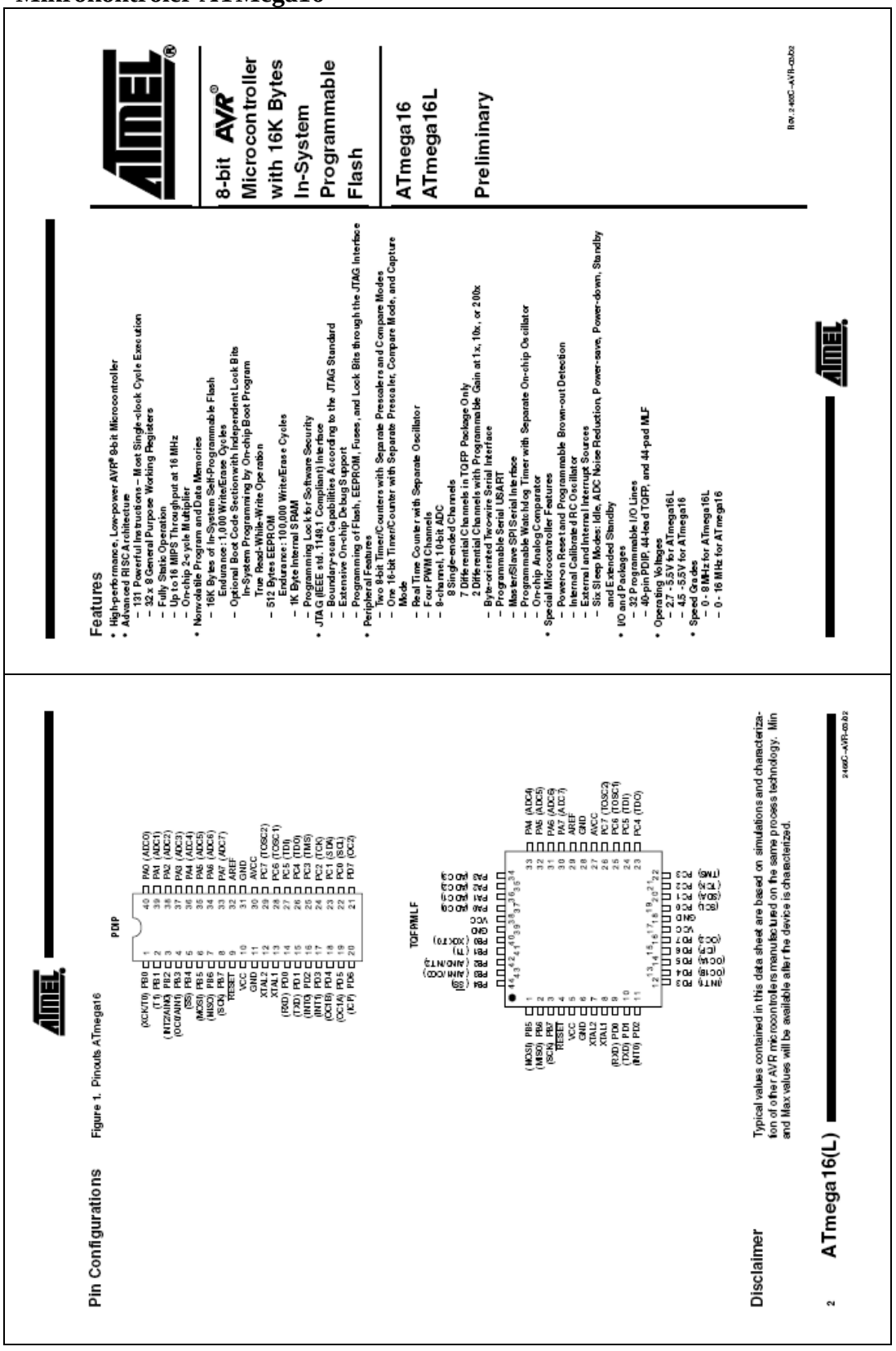
F = 1 : 5×10 dot matrix
F = 0 : 5×7 dot matrix

BF = 1 : Internal operation in progress
BF = 0 : Instruction can be accepted

* An execution time indicates maximum value when f_{ex} is 250 kHz. It changes at the inverse proportion of f_{ex}.

– 14 –

Mikrokontroler ATMega16



Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
  - 16K Bytes of In-System Self-Programmable Flash
    Endurance: 1,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 512 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 1K Byte Internal SRAM
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
    8 Single-ended Channels
    7 Differential Channels in TQFP Package Only
    2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
  - Byte-oriented Two-wire Serial Interface
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, and 44-pad MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega16L
  - 4.5 - 5.5V for ATmega16
- Speed Grades
  - 0 - 8 MHz for ATmega16L
  - 0 - 16 MHz for ATmega16

8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

ATmega16
ATmega16L

Preliminary

Rev. 2466G–AVR–05/02

Pin Configurations

Figure 1. Pinouts ATmega16

PDIP

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

TQFP/MLF

(MOSI) PB5 1 — 33 PA4 (ADC4)
(MISO) PB6 2 — 32 PA5 (ADC5)
(SCK) PB7 3 — 31 PA6 (ADC6)
RESET 4 — 30 PA7 (ADC7)
VCC 5 — 29 AREF
GND 6 — 28 GND
XTAL2 7 — 27 AVCC
XTAL1 8 — 26 PC7 (TOSC2)
(RXD) PD0 9 — 25 PC6 (TOSC1)
(TXD) PD1 10 — 24 PC5 (TDI)
(INT0) PD2 11 — 23 PC4 (TDO)

Disclaimer

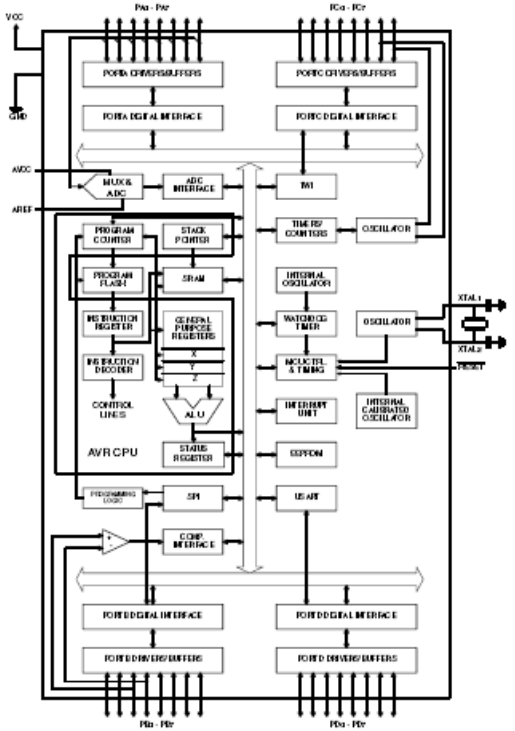
Typical values contained in this data sheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Overview

The ATmega16 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega16 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 2. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega16 provides the following features: 16K bytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes EEPROM, 1K byte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next External Interrupt or Hardware Reset. In Power-save mode, the Asynchronous Timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega16 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega16 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

Pin Descriptions

VCC — Digital supply voltage.

GND — Ground.

Port A (PA7..PA0) — Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

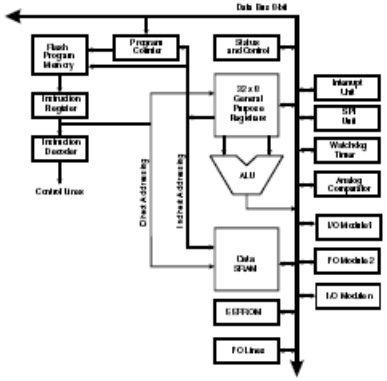
AVR CPU Core

Introduction

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Architectural Overview

Figure 3. Block Diagram of the AVR MCU Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

The fast-access Register file contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register file, the operation is executed, and the result is stored back in the Register file – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash Program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After

Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega16 as listed on page 55.

Port C (PC7..PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.

Port C also serves the functions of the JTAG interface and other special features of the ATmega16 as listed on page 58.

Port D (PD7..PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega16 as listed on page 60.

RESET

Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 35. Shorter pulses are not guaranteed to generate a reset.

XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting Oscillator amplifier.

AVCC

AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

AREF

AREF is the analog reference pin for the A/D Converter.

About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C Compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C Compiler documentation for more details.

an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section.

During interrupts and subroutine calls, the return address program counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The Stack Pointer SP is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the Status Register. All interrupts have a separate interrupt vector in the interrupt vector table. The interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register file, $20 - $5F.

ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the "Instruction Set" section for a detailed description.

Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

The AVR Status Register – SREG – is defined as:

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
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 – I: Global Interrupt Enable**

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

- **Bit 6 – T: Bit Copy Storage**

The Bit Copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source or destination for the operated bit. A bit from a register in the Register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register file by the BLD instruction.

- **Bit 5 – H: Half Carry Flag**

The Half Carry Flag H indicates a half carry in some arithmetic operations. Half Carry is useful in BCD arithmetic. See the "Instruction Set Description" for detailed information.

- **Bit 4 – S: Sign Bit, S = N ⊕ V**

The S-bit is always an exclusive or between the negative flag N and the two's complement overflow flag V. See the "Instruction Set Description" for detailed information.

- **Bit 3 – V: Two's Complement Overflow Flag**

The Two's Complement Overflow Flag V supports two's complement arithmetics. See the "Instruction Set Description" for detailed information.

- **Bit 2 – N: Negative Flag**

The Negative Flag N indicates a negative result in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

- **Bit 1 – Z: Zero Flag**

The Zero Flag Z indicates a zero result in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

- **Bit 0 – C: Carry Flag**

The Carry Flag C indicates a carry in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

General Purpose Register File

The Register File is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register file:

- One 8-bit output operand and one 8-bit result input
- Two 8-bit output operands and one 8-bit result input
- Two 8-bit output operands and one 16-bit result input
- One 16-bit output operand and one 16-bit result input

Figure 4 shows the structure of the 32 general purpose working registers in the CPU.

Stack Pointer

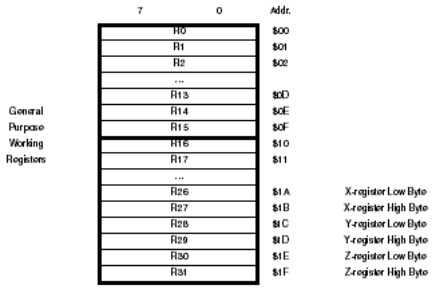
The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. The Stack Pointer Register always points to the top of the stack. Note that the stack is implemented as growing from higher memory locations to lower memory locations. This implies that a stack PUSH command decreases the Stack Pointer.

The Stack Pointer points to the data SRAM stack area where the Subroutine and Interrupt Stacks are located. This Stack space in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. The Stack Pointer must be set to point above $60. The Stack Pointer is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when the return address is pushed onto the Stack with subroutine call or interrupt. The Stack Pointer is incremented by one when data is popped from the Stack with the POP instruction, and it is incremented by two when data is popped from the Stack with return from subroutine RET or return from interrupt RETI.

The AVR Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.

Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	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	
	0	0	0	0	0	0	0	0	

Figure 4. AVR CPU General Purpose Working Registers



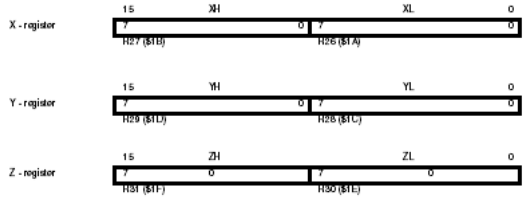
Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 4, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y-, and Z-pointer Registers can be set to index any register in the file.

The X-register, Y-register and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the Data Space. The three indirect address registers X, Y, and Z are defined as described in Figure 5.

Figure 5. The X-, Y-, and Z-registers



In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement (see the Instruction Set Reference for details).

I/O Memory

The I/O space definition of the ATmega16 is shown in "Register Summary" on page 301.

All ATmega16 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O Registers within the address range $00 - $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the Instruction Set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses $00 - $3F must be used. When addressing I/O Registers as data space using LD and ST instructions, $20 must be added to these addresses.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O Register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers $00 to $1F only.

The I/O and peripherals control registers are explained in later sections.

AVR ATmega16 Memories

This section describes the different memories in the ATmega16. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the ATmega16 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

In-System Reprogrammable Flash Program Memory

The ATmega16 contains 16K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 8K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Program section and Application Program section.

The Flash memory has an endurance of at least 1,000 write/erase cycles. The ATmega16 Program Counter (PC) is 13 bits wide, thus addressing the 8K program memory locations. The operation of Boot Program section and associated Boot Lock bits for software protection are described in detail in "Boot Loader Support – Read-While-Write Self-Programming" on page 241. "Memory Programming" on page 254 contains a detailed description on Flash data serial downloading using the SPI pins or the JTAG interface.

Constant tables can be allocated within the entire program memory address space (see the LPM – Load Program Memory Instruction Description).

Timing diagrams for instruction fetch and execution are presented in "Instruction Execution Timing" on page 11.

Figure 8. Program Memory Map

Interrupts

This section describes the specifics of the interrupt handling as performed in ATmega16. For a general explanation of the AVR interrupt handling, refer to "Reset and Interrupt Handling" on page 11.

Interrupt Vectors in ATmega16

Table 18. Reset and Interrupt Vectors

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$010	TIMER1 OVF	Timer/Counter1 Overflow
10	$012	TIMER0 OVF	Timer/Counter0 Overflow
11	$014	SPI, STC	Serial Transfer Complete
12	$016	USART, RXC	USART, Rx Complete
13	$018	USART, UDRE	USART Data Register Empty
14	$01A	USART, TXC	USART, Tx Complete
15	$01C	ADC	ADC Conversion Complete
16	$01E	EE_RDY	EEPROM Ready
17	$020	ANA_COMP	Analog Comparator
18	$022	TWI	Two-wire Serial Interface
19	$024	INT2	External Interrupt Request 2
20	$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	$028	SPM_RDY	Store Program Memory Ready

Notes: 1. When the BOOTRST fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming" on page 241.
2. When the IVSEL bit in GICR is set, interrupt vectors will be moved to the start of the Boot Flash section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash section.

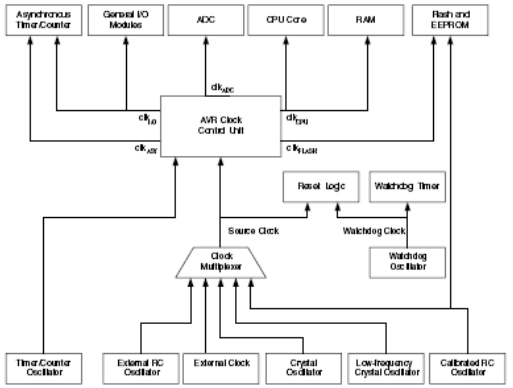
Table 19 shows Reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

System Clock and Clock Options

Clock Systems and their Distribution

Figure 11 presents the principal clock systems in the AVR and their distribution. All of the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes, as described in "Power Management and Sleep Modes" on page 30. The clock systems are detailed Figure 11.

Figure 11. Clock Distribution



CPU Clock – clk_{CPU}

The CPU clock is routed to parts of the system concerned with operation of the AVR core. Examples of such modules are the General Purpose Register File, the Status Register and the data memory holding the Stack Pointer. Halting the CPU clock inhibits the core from performing general operations and calculations.

I/O Clock – $clk_{I/O}$

The I/O clock is used by the majority of the I/O modules, like Timer/Counters, SPI, and USART. The I/O clock is also used by the External Interrupt module, but note that some external interrupts are detected by asynchronous logic, allowing such interrupts to be detected even if the I/O clock is halted. Also note that address recognition in the TWI module is carried out asynchronously when $clk_{I/O}$ is halted, enabling TWI address reception in all sleep modes.

Flash Clock – clk_{FLASH}

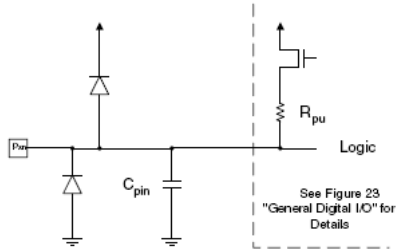
The Flash clock controls operation of the Flash interface. The Flash clock is usually active simultaneously with the CPU clock.

I/O Ports

Introduction

All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both V_{CC} and Ground as indicated in Figure 22. Refer to "Electrical Characteristics" on page 285 for a complete list of parameters.

Figure 22. I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case "x" represents the numbering letter for the port, and a lower case "n" represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. i.e., PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn. The physical I/O Registers and bit locations are listed in "Register Description for I/O Ports" on page 62.

Three I/O memory address locations are allocated for each port, one each for the Data Register – PORTx, Data Direction Register – DDRx, and the Port Input Pins – PINx. The Port Input Pins I/O location is read only, while the Data Register and the Data Direction Register are read/write. In addition, the Pull-up Disable – PUD bit in SFIOR disables the pull-up function for all pins in all ports when set.

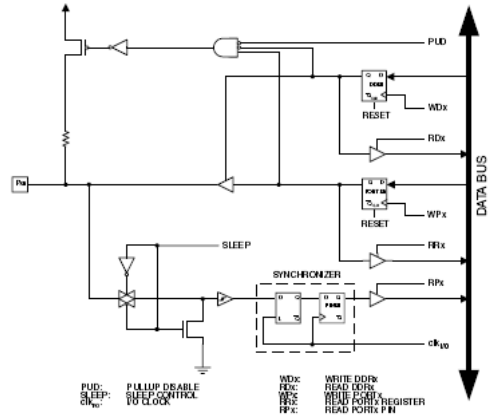
Using the I/O port as General Digital I/O is described in "Ports as General Digital I/O" on page 48. Most port pins are multiplexed with alternate functions for the peripheral features on the device. How each alternate function interferes with the port pin is described in "Alternate Port Functions" on page 52. Refer to the individual module sections for a full description of the alternate functions.

Note that enabling the alternate function of some of the port pins does not affect the use of the other pins in the port as general digital I/O.

Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 23 shows a functional description of one I/O-port pin, here generically called Pxn.

Figure 23. General Digital I/O[1]



Note: 1. WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in "Register Description for I/O Ports" on page 62, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx Register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when a reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled ({DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up

Table 21. Generic Description of Overriding Signals for Alternate Functions

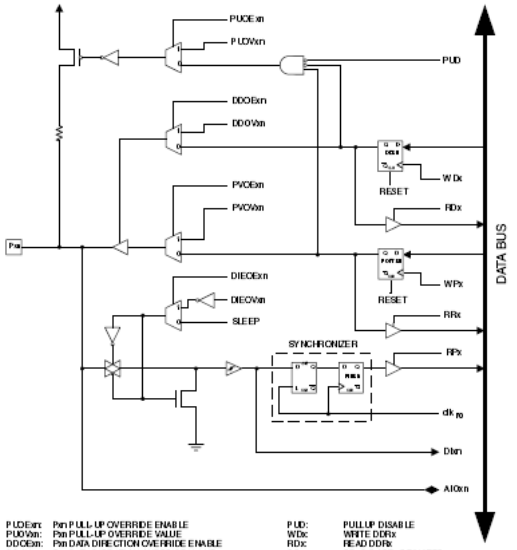
Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD Register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn Register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn Register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn Register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn Register bit.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU-state (Normal Mode, sleep modes).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal Mode, sleep modes).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the schmitt trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/ output	This is the Analog Input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

The following subsections shortly describe the alternate functions for each port, and relate the overriding signals to the alternate function. Refer to the alternate function description for further details.

Alternate Port Functions

Most port pins have alternate functions in addition to being General Digital I/Os. Figure 26 shows how the port pin control signals from the simplified Figure 23 can be overridden by alternate functions. The overriding signals may not be present in all port pins, but the figure serves as a generic description applicable to all port pins in the AVR microcontroller family.

Figure 26. Alternate Port Functions(1)



PUOExn: Pxn PULL-UP OVERRIDE ENABLE
PUOVxn: Pxn PULL-UP OVERRIDE VALUE
DDOExn: Pxn DATA DIRECTION OVERRIDE ENABLE
DDOVxn: Pxn DATA DIRECTION OVERRIDE VALUE
PVOExn: Pxn PORT VALUE OVERRIDE ENABLE
PVOVxn: Pxn PORT VALUE OVERRIDE VALUE
DIEOExn: Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE
DIEOVxn: Pxn DIGITAL INPUT-ENABLE OVERRIDEVALUE
SLEEP: SLEEP CONTROL

PUD: PULLUP DISABLE
WDx: WRITE DDRx
RDx: READ DDRx
RRx: READ PORTx REGISTER
WPx: WRITE PORTx
RPx: READ PORTx PIN
$clk_{I/O}$: I/O CLOCK
Dixn: DIGITAL INPUT PIN n ON PORTx
AIOxn: ANALOG INPUT/OUTPUT PIN nON PORTx

Note: 1. WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

Table 21 summarizes the function of the overriding signals. The pin and port indexes from Figure 26 are not shown in the succeeding tables. The overriding signals are generated internally in the modules having the alternate function.

Special Function I/O Register – SFIOR

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	ADHSM	ACME	PUD	PSR2	PSR10	SFIOR
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 2 – PUD: Pull-up disable**

When this bit is written to one, the pull-ups in the I/O ports are disabled even if the DDxn and PORTxn Registers are configured to enable the pull-ups ({DDxn, PORTxn} = 0b01). See "Configuring the Pin" on page 48 for more details about this feature.

Alternate Functions of Port A

Port A has an alternate function as analog input for the ADC as shown in Table 22. If some Port A pins are configured as outputs, it is essential that these do not switch when a conversion is in progress. This might corrupt the result of the conversion.

Table 22. Port A Pins Alternate Functions

Port Pin	Alternate Function
PA7	ADC7 (ADC input channel 7)
PA6	ADC6 (ADC input channel 6)
PA5	ADC5 (ADC input channel 5)
PA4	ADC4 (ADC input channel 4)
PA3	ADC3 (ADC input channel 3)
PA2	ADC2 (ADC input channel 2)
PA1	ADC1 (ADC input channel 1)
PA0	ADC0 (ADC input channel 0)

Table 23 and Table 24 relate the alternate functions of Port A to the overriding signals shown in Figure 26 on page 52.

Table 23. Overriding Signals for Alternate Functions in PA7..PA4

Signal Name	PA7/ADC7	PA6/ADC6	PA5/ADC5	PA4/ADC4
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AIO	ADC7 INPUT	ADC6 INPUT	ADC5 INPUT	ADC4 INPUT

Table 24. Overriding Signals for Alternate Functions in PA3..PA0

Signal Name	PA3/ADC3	PA2/ADC2	PA1/ADC1	PA0/ADC0
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AIO	ADC3 INPUT	ADC2 INPUT	ADC1 INPUT	ADC0 INPUT

Alternate Functions of Port B

The Port B pins with alternate functions are shown in Table 25.

Table 25. Port B Pins Alternate Functions

Port Pin	Alternate Functions
PB7	SCK (SPI Bus Serial Clock)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB4	$\overline{SS}$ (SPI Slave Select Input)
PB3	AIN1 (Analog Comparator Negative Input) OC0 (Timer/Counter0 Output Compare Match Output)
PB2	AIN0 (Analog Comparator Positive Input) INT2 (External Interrupt 2 Input)
PB1	T1 (Timer/Counter1 External Counter Input)
PB0	T0 (Timer/Counter0 External Counter Input) XCK (USART External Clock Input/Output)

The alternate pin configuration is as follows:

- **SCK – Port B, Bit 7**

SCK: Master Clock output, Slave Clock input pin for SPI channel. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB7. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB7. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB7 bit.

- **MISO – Port B, Bit 6**

MISO: Master Data input, Slave Data output pin for SPI channel. When the SPI is enabled as a Master, this pin is configured as an input regardless of the setting of DDB6. When the SPI is enabled as a Slave, the data direction of this pin is controlled by

DDB6. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB6 bit.

- **MOSI – Port B, Bit 5**

MOSI: SPI Master Data output, Slave Data input for SPI channel. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB5. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB5. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB5 bit.

- **$\overline{SS}$ – Port B, Bit 4**

$\overline{SS}$: Slave Select input. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB4. As a Slave, the SPI is activated when this pin is driven low. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB4. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB4 bit.

- **AIN1/OC0 – Port B, Bit 3**

AIN1, Analog Comparator Negative Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the analog comparator.

OC0, Output Compare Match output: The PB3 pin can serve as an external output for the Timer/Counter0 Compare Match. The PB3 pin has to be configured as an output (DDB3 set (one)) to serve this function. The OC0 pin is also the output pin for the PWM mode timer function.

- **AIN0/INT2 – Port B, Bit 2**

AIN0, Analog Comparator Positive input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the Analog Comparator.

INT2, External Interrupt Source 2: The PB2 pin can serve as an external interrupt source to the MCU.

- **T1 – Port B, Bit 1**

T1, Timer/Counter1 Counter Source.

- **T0/XCK – Port B, Bit 0**

T0, Timer/Counter0 Counter Source.

XCK, USART External Clock. The Data Direction Register (DDB0) controls whether the clock is output (DDB0 set) or input (DDB0 cleared). The XCK pin is active only when the USART operates in Synchronous mode.

Table 26 and Table 27 relate the alternate functions of Port B to the overriding signals shown in Figure 26 on page 52. SPI MSTR INPUT and SPI SLAVE OUTPUT constitute the MISO signal, while MOSI is divided into SPI MSTR OUTPUT and SPI SLAVE INPUT.

Table 26. Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/SCK	PB6/MISO	PB5/MOSI	PB4/$\overline{SS}$
PUOE	SPE • $\overline{MSTR}$	SPE • MSTR	SPE • $\overline{MSTR}$	SPE • $\overline{MSTR}$
PUOV	PORTB7 • $\overline{PUD}$	PORTB6 • $\overline{PUD}$	PORTB5 • $\overline{PUD}$	PORTB4 • $\overline{PUD}$
DDOE	SPE • $\overline{MSTR}$	SPE • MSTR	SPE • $\overline{MSTR}$	SPE • $\overline{MSTR}$
DDOV	0	0	0	0
PVOE	SPE • MSTR	SPE • $\overline{MSTR}$	SPE • MSTR	0
PVOV	SCK OUTPUT	SPI SLAVE OUTPUT	SPI MSTR OUTPUT	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	SCK INPUT	SPI MSTR INPUT	SPI SLAVE INPUT	SPI $\overline{SS}$
AIO	–	–	–	–

Table 27. Overriding Signals for Alternate Functions in PB3..PB0

Signal Name	PB3/OC0/AIN1	PB2/INT2/AIN0	PB1/T1	PB0/T0/XCK
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	OC0 ENABLE	0	0	UMSEL
PVOV	OC0	0	0	XCK OUTPUT
DIEOE	0	INT2 ENABLE	0	0
DIEOV	0	1	0	0
DI	–	INT2 INPUT	T1 INPUT	XCK INPUT/T0 INPUT
AIO	AIN1 INPUT	AIN0 INPUT	–	–

Table 30. Overriding Signals for Alternate Functions in PC3..PC0[1]

Signal Name	PC3/TMS	PC2/TCK	PC1/SDA	PC0/SCL
PUOE	JTAGEN	JTAGEN	TWEN	TWEN
PUOV	1	1	PORTC1 • $\overline{PUD}$	PORTC0 • $\overline{PUD}$
DDOE	JTAGEN	JTAGEN	TWEN	TWEN
DDOV	0	0	SDA_OUT	SCL_OUT
PVOE	0	0	TWEN	TWEN
PVOV	0	0	0	0
DIEOE	JTAGEN	JTAGEN	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AIO	TMS	TCK	SDA INPUT	SCL INPUT

Note: 1. When enabled, the Two-wire Serial Interface enables slew-rate controls on the output pins PC0 and PC1. This is not shown in the figure. In addition, spike filters are connected between the AIO outputs shown in the port figure and the digital logic of the TWI module.

Alternate Functions of Port D

The Port D pins with alternate functions are shown in Table 31.

Table 31. Port D Pins Alternate Functions

Port Pin	Alternate Function
PD7	OC2 (Timer/Counter2 Output Compare Match Output)
PD6	ICP (Timer/Counter1 Input Capture Pin)
PD5	OC1A (Timer/Counter1 Output Compare A Match Output)
PD4	OC1B (Timer/Counter1 Output Compare B Match Output)
PD3	INT1 (External Interrupt 1 Input)
PD2	INT0 (External Interrupt 0 Input)
PD1	TXD (USART Output Pin)
PD0	RXD (USART Input Pin)

The alternate pin configuration is as follows:

- **OC2 – Port D, Bit 7**

OC2, Timer/Counter2 Output Compare Match output: The PD7 pin can serve as an external output for the Timer/Counter2 Output Compare. The pin has to be configured as an output (DDD7 set (one)) to serve this function. The OC2 pin is also the output pin for the PWM mode timer function.

- **ICP – Port D, Bit 6**

ICP – Input Capture Pin: The PD6 pin can act as an Input Capture pin for Timer/Counter1.

Alternate Functions of Port C

The Port C pins with alternate functions are shown in Table 28. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.

Table 28. Port C Pins Alternate Functions

Port Pin	Alternate Function
PC7	TOSC2 (Timer Oscillator Pin 2)
PC6	TOSC1 (Timer Oscillator Pin 1)
PC5	TDI (JTAG Test Data In)
PC4	TDO (JTAG Test Data Out)
PC3	TMS (JTAG Test Mode Select)
PC2	TCK (JTAG Test Clock)
PC1	SDA (Two-wire Serial Bus Data Input/Output Line)
PC0	SCL (Two-wire Serial Bus Clock Line)

The alternate pin configuration is as follows:

- **TOSC2 – Port C, Bit 7**

TOSC2, Timer Oscillator pin 2: When the AS2 bit in ASSR is set (one) to enable asynchronous clocking of Timer/Counter2, pin PC7 is disconnected from the port, and becomes the inverting output of the Oscillator amplifier. In this mode, a Crystal Oscillator is connected to this pin, and the pin can not be used as an I/O pin.

- **TOSC1 – Port C, Bit 6**

TOSC1, Timer Oscillator pin 1: When the AS2 bit in ASSR is set (one) to enable asynchronous clocking of Timer/Counter2, pin PC6 is disconnected from the port, and becomes the input of the inverting Oscillator amplifier. In this mode, a Crystal Oscillator is connected to this pin, and the pin can not be used as an I/O pin.

- **TDI – Port C, Bit 5**

TDI, JTAG Test Data In: Serial input data to be shifted in to the Instruction Register or Data Register (scan chains). When the JTAG interface is enabled, this pin can not be used as an I/O pin.

- **TDO – Port C, Bit 4**

TDO, JTAG Test Data Out: Serial output data from Instruction register or Data Register. When the JTAG interface is enabled, this pin can not be used as an I/O pin.

- **TMS – Port C, Bit 3**

TMS, JTAG Test Mode Select: This pin is used for navigating through the TAP-controller state machine. When the JTAG interface is enabled, this pin can not be used as an I/O pin.

- **TCK – Port C, Bit 2**

TCK, JTAG Test Clock: JTAG operation is synchronous to TCK. When the JTAG interface is enabled, this pin can not be used as an I/O pin.

Timer/Counter0 and Timer/Counter1 Prescalers

Timer/Counter1 and Timer/Counter0 share the same prescaler module, but the Timer/Counters can have different prescaler settings. The description below applies to both Timer/Counter1 and Timer/Counter0.

Internal Clock Source

The Timer/Counter can be clocked directly by the system clock (by setting the CSn2:0 = 1). This provides the fastest operation, with a maximum Timer/Counter clock frequency equal to system clock frequency ($f_{CLK\_I/O}$). Alternatively, one of four taps from the prescaler can be used as a clock source. The prescaled clock has a frequency of either $f_{CLK\_I/O}/8$, $f_{CLK\_I/O}/64$, $f_{CLK\_I/O}/256$, or $f_{CLK\_I/O}/1024$.

Prescaler Reset

The prescaler is free running, i.e., operates independently of the clock select logic of the Timer/Counter, and it is shared by Timer/Counter1 and Timer/Counter0. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler (6 > CSn2:0 > 1). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

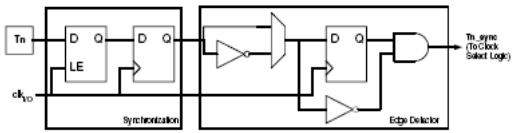
It is possible to use the Prescaler Reset for synchronizing the Timer/Counter to program execution. However, care must be taken if the other Timer/Counter that shares the same prescaler also uses prescaling. A prescaler reset will affect the prescaler period for all Timer/Counters it is connected to.

External Clock Source

An external clock source applied to the T1/T0 pin can be used as Timer/Counter clock (clk_{T1}/clk_{T0}). The T1/T0 pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 38 shows a functional equivalent block diagram of the T1/T0 synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ($clk_{I/O}$). The latch is transparent in the high period of the internal system clock.

The edge detector generates one clk_{T1}/clk_{T0} pulse for each positive (CSn2:0 = 7) or negative (CSn2:0 = 6) edge it detects.

Figure 38. T1/T0 Pin Sampling



The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the T1/T0 pin to the counter is updated.

Enabling and disabling of the clock input must be done when T1/T0 has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

Each half period of the external clock applied must be longer than one system clock cycle to ensure correct sampling. The external clock must be guaranteed to have less

- **OC1A – Port D, Bit 5**

OC1A, Output Compare Match A output: The PD5 pin can serve as an external output for the Timer/Counter1 Output Compare A. The pin has to be configured as an output (DDD5 set (one)) to serve this function. The OC1A pin is also the output pin for the PWM mode timer function.

- **OC1B – Port D, Bit 4**

OC1B, Output Compare Match B output: The PD4 pin can serve as an external output for the Timer/Counter1 Output Compare B. The pin has to be configured as an output (DDD4 set (one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.

- **INT1 – Port D, Bit 3**

INT1, External Interrupt Source 1: The PD3 pin can serve as an external interrupt source.

- **INT0 – Port D, Bit 2**

INT0, External Interrupt Source 0: The PD2 pin can serve as an external interrupt source.

- **TXD – Port D, Bit 1**

TXD, Transmit Data (Data output pin for the USART). When the USART Transmitter is enabled, this pin is configured as an output regardless of the value of DDD1.

- **RXD – Port D, Bit 0**

RXD, Receive Data (Data input pin for the USART). When the USART Receiver is enabled this pin is configured as an input regardless of the value of DDD0. When the USART forces this pin to be an input, the pull-up can still be controlled by the PORTD0 bit.

Table 32 and Table 33 relate the alternate functions of Port D to the overriding signals shown in Figure 26 on page 52.

Table 32. Overriding Signals for Alternate Functions PD7..PD4

Signal Name	PD7/OC2	PD6/ICP	PD5/OC1A	PD4/OC1B
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	OC2 ENABLE	0	OC1A ENABLE	OC1B ENABLE
PVOV	OC2	0	OC1A	OC1B
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	ICP INPUT	–	–
AIO	–	–	–	–

16-bit Timer/Counter1

The 16-bit Timer/Counter unit allows accurate program execution timing (event management), wave generation, and signal timing measurement. The main features are:

- True 16-bit Design (i.e., Allows 16-bit PWM)
- Two Independent Output Compare Units
- Double Buffered Output Compare Registers
- One Input Capture Unit
- Input Capture Noise Canceler
- Clear Timer on Compare Match (Auto Reload)
- Glitch-free, Phase Correct Pulse Width Modulator (PWM)
- Variable PWM Period
- Frequency Generator
- External Event Counter
- Four Independent Interrupt Sources (TOV1, OCF1A, OCF1B, and ICF1)

Overview

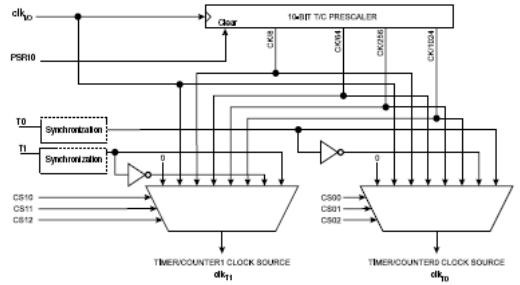
Most register and bit references in this document are written in general form. A lower case "n" replaces the Timer/Counter number, and a lower case "x" replaces the output compare unit channel. However, when using the register or bit defines in a program, the precise form must be used (i.e., TCNT1 for accessing Timer/Counter1 counter value and so on). The physical I/O Register and bit locations for ATmega16 are listed in the "16-bit Timer/Counter Register Description" on page 104.

A simplified block diagram of the 16-bit Timer/Counter is shown in Figure 40. CPU accessible I/O Registers, including I/O bits and I/O pins, are shown in bold.

than half the system clock frequency ($f_{ExtClk} < f_{clk\_I/O}/2$) given a 50/50% duty cycle. Since the edge detector uses sampling, the maximum frequency of an external clock it can detect is half the sampling frequency (Nyquist sampling theorem). However, due to variation of the system clock frequency and duty cycle caused by Oscillator source (crystal, resonator, and capacitors) tolerances, it is recommended that maximum frequency of an external clock source is less than $f_{clk\_I/O}/2.5$.

An external clock source can not be prescaled.

Figure 39. Prescaler for Timer/Counter0 and Timer/Counter1(1)



Note: 1. The synchronization logic on the input pins (T1/T0) is shown in Figure 38.

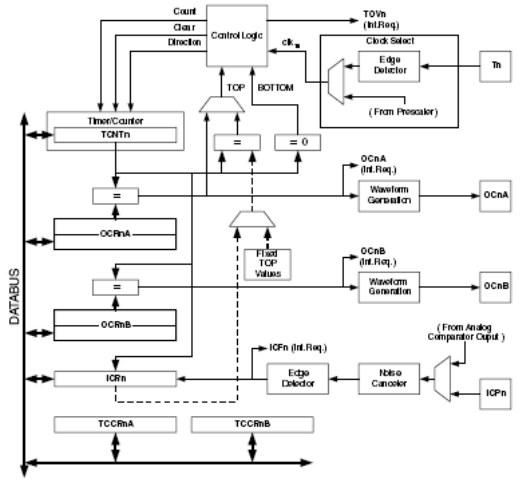
Special Function IO Register – SFIOR

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	ADHSM	ACME	PUD	PSR2	PSR10	SFIOR
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 0 – PSR10: Prescaler Reset Timer/Counter1 and Timer/Counter0**

When this bit is written to one, the Timer/Counter1 and Timer/Counter0 prescaler will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. Note that Timer/Counter1 and Timer/Counter0 share the same prescaler and a reset of this prescaler will affect both timers. This bit will always be read as zero.

Figure 40. 16-bit Timer/Counter Block Diagram[1]



Note: 1. Refer to Figure 1 on page 2, Table 25 on page 55, and Table 31 on page 60 for Timer/Counter1 pin placement and description.

Registers

The *Timer/Counter* (TCNT1), *Output Compare Registers* (OCR1A/B), and *Input Capture Register* (ICR1) are all 16-bit registers. Special procedures must be followed when accessing the 16-bit registers. These procedures are described in the section "Accessing 16-bit Registers" on page 86. The *Timer/Counter Control Registers* (TCCR1A/B) are 8-bit registers and have no CPU access restrictions. Interrupt requests (abbreviated to Int.Req. in the figure) signals are all visible in the *Timer Interrupt Flag Register* (TIFR). All interrupts are individually masked with the *Timer Interrupt Mask Register* (TIMSK). TIFR and TIMSK are not shown in the figure since these registers are shared by other timer units.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T1 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the clock select logic is referred to as the timer clock (clk_{T1}).

The double buffered Output Compare Registers (OCR1A/B) are compared with the Timer/Counter value at all time. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pin

(OC1A/B). See "Output Compare Units" on page 91. The compare match event will also set the Compare Match Flag (OCF1A/B) which can be used to generate an output compare interrupt request.

The Input Capture Register can capture the Timer/Counter value at a given external (edge triggered) event on either the Input Capture Pin (ICP1) or on the Analog Comparator pins (See "Analog Comparator" on page 195.) The input capture unit includes a digital filtering unit (Noise Canceler) for reducing the chance of capturing noise spikes.

The TOP value, or maximum Timer/Counter value, can in some modes of operation be defined by either the OCR1A Register, the ICR1 Register, or by a set of fixed values. When using OCR1A as TOP value in a PWM mode, the OCR1A Register can not be used for generating a PWM output. However, the TOP value will in this case be double buffered allowing the TOP value to be changed in run time. If a fixed TOP value is required, the ICR1 Register can be used as an alternative, freeing the OCR1A to be used as PWM output.

Definitions

The following definitions are used extensively throughout the document:

Table 43. Definitions

BOTTOM	The counter reaches the *BOTTOM* when it becomes 0x0000.
MAX	The counter reaches its *MAX*imum when it becomes 0xFFFF (decimal 65535).
TOP	The counter reaches the *TOP* when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be one of the fixed values: 0x00FF, 0x01FF, or 0x03FF, or to the value stored in the OCR1A or ICR1 register. The assignment is dependent of the mode of operation.

Compatibility

The 16-bit Timer/Counter has been updated and improved from previous versions of the 16-bit AVR Timer/Counter. This 16-bit Timer/Counter is fully compatible with the earlier version regarding:

- All 16-bit Timer/Counter related I/O Register address locations, including timer interrupt registers.
- Bit locations inside all 16-bit Timer/Counter Registers, including timer interrupt registers.
- Interrupt Vectors.

The following control bits have changed name, but have same functionality and register location:

- PWM10 is changed to WGM10.
- PWM11 is changed to WGM11.
- CTC1 is changed to WGM12.

The following bits are added to the 16-bit Timer/Counter Control Registers:

- FOC1A and FOC1B are added to TCCR1A.
- WGM13 is added to TCCR1B.

The 16-bit Timer/Counter has improvements that will affect the compatibility in some special cases.

USART

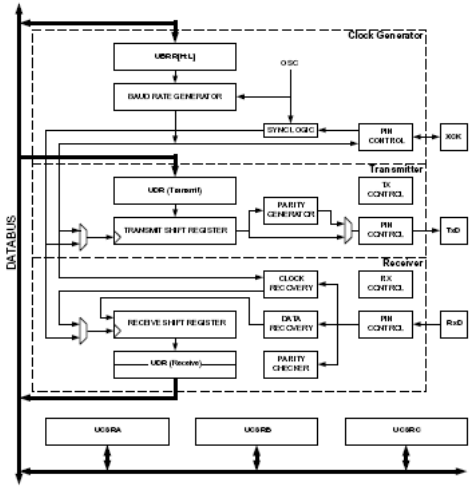
The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device. The main features are:

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous Operation
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty, and RX Complete
- Multi-processor Communication Mode
- Double Speed Asynchronous Communication Mode

Overview

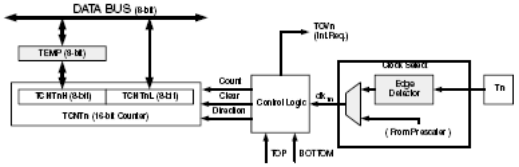
A simplified block diagram of the USART transmitter is shown in Figure 69. CPU accessible I/O Registers and I/O pins are shown in bold.

Figure 69. USART Block Diagram[1]



Note: 1. Refer to Figure 1 on page 2, Table 33 on page 62, and Table 27 on page 57 for USART pin placement.

Figure 41. Counter Unit Block Diagram



Signal description (internal signals):

Count	Increment or decrement TCNT1 by 1.
Direction	Select between increment and decrement.
Clear	Clear TCNT1 (set all bits to zero).
clk_{T1}	Timer/Counter clock.
TOP	Signalize that TCNT1 has reached maximum value.
BOTTOM	Signalize that TCNT1 has reached minimum value (zero).

The 16-bit counter is mapped into two 8-bit I/O memory locations: *Counter High* (TCNT1H) containing the upper eight bits of the counter, and *Counter Low* (TCNT1L) containing the lower 8 bits. The TCNT1H Register can only be indirectly accessed by the CPU. When the CPU does an access to the TCNT1H I/O location, the CPU accesses the high byte temporary register (TEMP). The temporary register is updated with the TCNT1H value when the TCNT1L is read, and TCNT1H is updated with the temporary register value when TCNT1L is written. This allows the CPU to read or write the entire 16-bit counter value within one clock cycle via the 8-bit data bus. It is important to notice that there are special cases of writing to the TCNT1 Register when the counter is counting that will give unpredictable results. The special cases are described in the sections where they are of importance.

Depending on the mode of operation used, the counter is cleared, incremented, or decremented at each *timer clock* (clk_{T1}). The clk_{T1} can be generated from an external or internal clock source, selected by the *Clock Select* bits (CS12:0). When no clock source is selected (CS12:0 = 0) the timer is stopped. However, the TCNT1 value can be accessed by the CPU, independent of whether clk_{T1} is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the *Waveform Generation Mode* bits (WGM13:0) located in the *Timer/Counter Control Registers* A and B (TCCR1A and TCCR1B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare outputs OC1x. For more details about advanced counting sequences and waveform generation, see "Modes of Operation" on page 94.

The *Timer/Counter Overflow* (TOV1) flag is set according to the mode of operation selected by the WGM13:0 bits. TOV1 can be used for generating a CPU interrupt.

The dashed boxes in the block diagram separate the three main parts of the USART (listed from the top): Clock Generator, Transmitter and Receiver. Control registers are shared by all units. The clock generation logic consists of synchronization logic for external clock input used by synchronous slave operation, and the baud rate generator. The XCK (Transfer Clock) pin is only used by Synchronous Transfer mode. The Transmitter consists of a single write buffer, a serial shift register, parity generator and control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The Receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery units are used for asynchronous data reception. In addition to the recovery units, the receiver includes a parity checker, control logic, a Shift Register and a two level receive buffer (UDR). The receiver supports the same frame formats as the transmitter, and can detect frame error, data overrun and parity errors.

AVR USART vs. AVR UART – Compatibility

The USART is fully compatible with the AVR UART regarding:

- Bit locations inside all USART Registers
- Baud Rate Generation
- Transmitter Operation
- Transmit Buffer Functionality
- Receiver Operation

However, the receive buffering has two improvements that will affect the compatibility in some special cases:

- A second buffer register has been added. The two buffer registers operate as a circular FIFO buffer. Therefore the UDR must only be read once for each incoming data! More important is the fact that the error flags (FE and DOR) and the 9th data bit (RXB8) are buffered with the data in the receive buffer. Therefore the status bits must always be read before the UDR Register is read. Otherwise the error status will be lost since the buffer state is lost.
- The receiver Shift Register can now act as a third buffer level. This is done by allowing the received data to remain in the serial Shift Register (see Figure 69) if the buffer registers are full, until a new start bit is detected. The USART is therefore more resistant to Data OverRun (DOR) error conditions.

The following control bits have changed name, but have same functionality and register location:

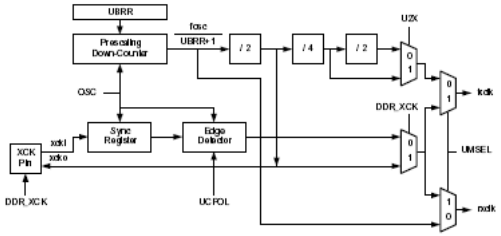
- CHR9 is changed to UCSZ2
- OR is changed to DOR

Clock Generation

The clock generation logic generates the base clock for the Transmitter and Receiver. The USART supports four modes of clock operation: Normal Asynchronous, Double Speed Asynchronous, Master Synchronous and Slave Synchronous mode. The UMSEL bit in USART Control and Status Register C (UCSRC) selects between asynchronous and synchronous operation. Double Speed (Asynchronous mode only) is controlled by the U2X found in the UCSRA Register. When using Synchronous mode (UMSEL = 1), the Data Direction Register for the XCK pin (DDR_XCK) controls whether the clock source is internal (Master mode) or external (Slave mode). The XCK pin is only active when using Synchronous mode.

Figure 70 shows a block diagram of the clock generation logic.

Figure 70. Clock Generation Logic, Block Diagram



Signal description:

txclk	Transmitter clock (Internal Signal).
rxclk	Receiver base clock (Internal Signal).
xcki	Input from XCK pin (Internal Signal). Used for synchronous slave operation.
xcko	Clock output to XCK pin (Internal Signal). Used for synchronous master operation.
fosc	XTAL pin frequency (System Clock).

Internal Clock Generation – The Baud Rate Generator

Internal clock generation is used for the asynchronous and the synchronous master modes of operation. The description in this section refers to Figure 70.

The USART Baud Rate Register (UBRR) and the down-counter connected to it function as a programmable prescaler or baud rate generator. The down-counter, running at system clock (fosc), is loaded with the UBRR value each time the counter has counted down to zero or when the UBRRL Register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output (= fosc/(UBRR+1)). The Transmitter divides the baud rate generator clock output by 2, 8 or 16 depending on mode. The baud rate generator output is used directly by the receiver's clock and data recovery units. However, the recovery units use a state machine that uses 2, 8 or 16 states depending on mode set by the state of the UMSEL, U2X and DDR_XCK bits.

Table 60 contains equations for calculating the baud rate (in bits per second) and for calculating the UBRR value for each mode of operation using an internally generated clock source.

Table 60. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{OSC}}{16(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{OSC}}{8(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{OSC}}{2(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{2BAUD} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps).

BAUD Baud rate (in bits per second, bps)

f_{OSC} System Oscillator clock frequency

UBRR Contents of the UBRRH and UBRRL Registers, (0 - 4095)

Some examples of UBRR values for some system clock frequencies are found in Table 68 (see page 162).

Double Speed Operation (U2X)

The transfer rate can be doubled by setting the U2X bit in UCSRA. Setting this bit only has effect for the asynchronous operation. Set this bit to zero when using synchronous operation.

Setting this bit will reduce the divisor of the baud rate divider from 16 to 8, effectively doubling the transfer rate for asynchronous communication. Note however that the receiver will in this case only use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery, and therefore a more accurate baud rate setting and system clock are required when this mode is used. For the Transmitter, there are no downsides.

External Clock

External clocking is used by the synchronous slave modes of operation. The description in this section refers to Figure 70 for details.

External clock input from the XCK pin is sampled by a synchronization register to minimize the chance of meta-stability. The output from the synchronization register must then pass through an edge detector before it can be used by the Transmitter and receiver. This process introduces a two CPU clock period delay and therefore the maximum external XCK clock frequency is limited by the following equation:

$$f_{XCK} < \frac{f_{OSC}}{4}$$

Note that f_{osc} depends on the stability of the system clock source. It is therefore recommended to add some margin to avoid possible loss of data due to frequency variations.

Synchronous Clock Operation

When Synchronous mode is used (UMSEL = 1), the XCK pin will be used as either clock input (Slave) or clock output (Master). The dependency between the clock edges and data sampling or data change is the same. The basic principle is that data input (on RxD) is sampled at the opposite XCK clock edge of the edge the data output (TxD) is changed.

The USART Character SiZe (UCSZ2:0) bits select the number of data bits in the frame. The USART Parity mode (UPM1:0) bits enable and set the type of parity bit. The selection between one or two stop bits is done by the USART Stop Bit Select (USBS) bit. The receiver ignores the second stop bit. An FE (Frame Error) will therefore only be detected in the cases where the first stop bit is zero.

Parity Bit Calculation

The parity bit is calculated by doing an exclusive-or of all the data bits. If odd parity is used, the result of the exclusive or is inverted. The relation between the parity bit and data bits is as follows::

$$P_{even} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$
$$P_{odd} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

P_{even} Parity bit using even parity

P_{odd} Parity bit using odd parity

d_n Data bit n of the character

If used, the parity bit is located between the last data bit and first stop bit of a serial frame.