LAMPIRAN - A

Instruksi Pengontrol Mikro

```
/*****************************************************
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 :
Version :
Date    : 8/5/2008
Author  : Piter
Company : lab
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>

#asm
   .equ __lcd_port=0x18 ;PORTB
#endasm
#include <lcd.h>
#include <IP.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];
```

```
#if RX_BUFFER_SIZE<256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

bit rx_buffer_overflow;

interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index]=data;
   if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      };
   };

        if (data == 13)
        {
        i=0;
        for (iw=0;iw<30;iw++)tdata2[iw]=tdata[iw];
        for (iw=0;iw<30;iw++)tdata[iw]=0;
        }
tdata[i]=data;
i++;
}

#ifndef _DEBUG_TERMINAL_IO_
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
```

```
#pragma used-
#endif

interrupt [TIM1_COMPA] void timer1_compa_isr(void)
{
for (iw=0;iw<9;iw++) pos[iw]=0;
for (iw=0;iw<30;iw++)tdata3[iw]=tdata2[iw];
pos_pos=0;
for (iw=0;iw<30;iw++)
{
if(tdata3[iw]==32)pos_pos++;
if(tdata3[iw+1]!=32)pos[pos_pos]=(pos[pos_pos]*10)+ (tdata3[iw+1]-48);
}
}

void main(void)
{
TCCR1A=0x00;
TCCR1B=0x0B;
OCR1AH=0x08;
OCR1AL=0x70;

TIMSK=0x10;
UCSRA=0x00;
UCSRB=0x98;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;
ACSR=0x80;
SFIOR=0x00;
DDRD.2=1;
DDRD.3=1;
DDRD.4=1;
DDRD.5=1;
DDRD.6=1;
DDRD.7=1;



lcd_init(16);
#asm("sei")

printf("L0 1\r");
delay_ms(200);
printf("L0 0\r");
delay_ms(200);
```

```
CmuCamInit();

while (1)
     {


//


       for (iw1=0;iw1<10;iw1++)
       {
       lcd_putsf("A>SET JARAK\n");
       lcd_putsf("B>SET WARNA ");

       if(scan_keypad()=='A')
       {
       lcd_clear();

jarak = keytonum();

       delay_ms(1000);
       lcd_clear();
       };

       if(scan_keypad()=='B')
       {
       lcd_clear();
       delay_ms(1000);



       for (iw=0;iw<30;iw++)
       {
       lcd_putsf("A>MERAH\n");
       lcd_putsf("B>BIRU ");

       if(scan_keypad()=='A')
       {
lcd_clear();
printf("RS\r");
delay_ms(200);
printf("L0 1\r");
delay_ms(200);
printf("L0 0\r");
delay_ms(200);
printf("TC 85 135 8 58 0 41\r");
delay_ms(200);
       lcd_putsf("TRACK \nWARNA MERAH");
```

```
        delay_ms(500);
        lcd_clear();

        };
        if(scan_keypad()=='B')
        {
lcd_clear();
printf("RS\r");
delay_ms(200);
printf("L0 1\r");
delay_ms(200);
printf("L0 0\r");
delay_ms(200);
printf("TC 0 41 54 104 63 113\r");
delay_ms(200);
        lcd_putsf("TRACK \nWARNA BIRU");
        delay_ms(500);
        lcd_clear();
        };


        delay_ms(150);
        lcd_clear();
        }
        for (iw=0;iw<30;iw++)
        {
        lcd_putsf("C>KUNING\n");
        if(scan_keypad()=='C')
        {
lcd_clear();
printf("RS\r");
delay_ms(200);
printf("L0 1\r");
delay_ms(200);
printf("L0 0\r");
delay_ms(200);
printf("TC 86 136 137 187 0 43\r");
delay_ms(200);
        lcd_putsf("TRACK \nWARNA KUNING");
        delay_ms(500);
        lcd_clear();
        };
        delay_ms(150);
        lcd_clear();
        }
        delay_ms(2500);
        lcd_clear();
        };
```

```
delay_ms(150);
lcd_clear();
}

for (iw1=0;iw1<10;iw1++)
{
lcd_putsf("C>RUN\n");
lcd_putsf("D>RESET");

if(scan_keypad()=='C')
{
lcd_clear();
while (scan_keypad()!= '#')
{
sprintf(lcd_buff,"X%d Y%d P%d",pos[1],pos[2],pos[6]);

if (((pos[1]<60)&&(pos[1]>40))&&(pos[6]<jarak))//lurus
        {
        PORTD.2=1;
        PORTD.3=0;
        PORTD.4=1;
        PORTD.5=1;
        PORTD.6=1;
        PORTD.7=0;
        delay_ms(300);
        PORTD.2=0;
        PORTD.3=0;
        PORTD.4=0;
        PORTD.5=0;
        PORTD.6=0;
        PORTD.7=0;
        };

if (((pos[1]<150)&&(pos[1]>60))&&(pos[6]<jarak)) //kiri
        {
        PORTD.2=0;
        PORTD.3=0;
        PORTD.4=1;
        PORTD.5=1;
        PORTD.6=1;
        PORTD.7=0;
        delay_ms(300);
        PORTD.2=0;
        PORTD.3=0;
        PORTD.4=0;
        PORTD.5=0;
        PORTD.6=0;
```

```
        PORTD.7=0;
        };
if (((pos[1]<40)&&(pos[1]>10))&&(pos[6]<jarak)) //kanan
        {
        PORTD.2=1;
        PORTD.3=0;
        PORTD.4=1;
        PORTD.5=1;
        PORTD.6=0;
        PORTD.7=0;
        delay_ms(300);
        PORTD.2=0;
        PORTD.3=0;
        PORTD.4=0;
        PORTD.5=0;
        PORTD.6=0;
        PORTD.7=0;
        };



if (((pos[1]<60)&&(pos[1]>40))&&(pos[6]>(jarak+15)))//mundur lurus
        {
        PORTD.2=0;
        PORTD.3=1;
        PORTD.4=1;
        PORTD.5=1;
        PORTD.6=0;
        PORTD.7=1;
        delay_ms(300);
        PORTD.2=0;
        PORTD.3=0;
        PORTD.4=0;
        PORTD.5=0;
        PORTD.6=0;
        PORTD.7=0;
        };

if (((pos[1]<150)&&(pos[1]>60))&&(pos[6]>(jarak+15))) //mundur kiri
        {
        PORTD.2=0;
        PORTD.3=1;
        PORTD.4=1;
        PORTD.5=1;
        PORTD.6=0;
        PORTD.7=0;
        delay_ms(300);
        PORTD.2=0;
```

```
            PORTD.3=0;
            PORTD.4=0;
            PORTD.5=0;
            PORTD.6=0;
            PORTD.7=0;
            };
      if (((pos[1]<40)&&(pos[1]>10))&&(pos[6]>(jarak+15))) //mundur kanan
            {
            PORTD.2=0;
            PORTD.3=0;
            PORTD.4=1;
            PORTD.5=1;
            PORTD.6=0;
            PORTD.7=1;
            delay_ms(300);
            PORTD.2=0;
            PORTD.3=0;
            PORTD.4=0;
            PORTD.5=0;
            PORTD.6=0;
            PORTD.7=0;
            };


      lcd_puts(lcd_buff);
      delay_ms(100);
      lcd_clear();
      }

      delay_ms(1000);
      lcd_clear();
      };

      if(scan_keypad()=='D')
      {
      lcd_clear();
      lcd_putsf("RESET");
printf("RS\r");
delay_ms(200);
printf("L0 1\r");
delay_ms(200);
printf("L0 0\r");
delay_ms(200);
      delay_ms(1000);
      lcd_clear();
      };

      delay_ms(150);
```

```
        lcd_clear();
        }

      };
}
```

LAMPIRAN - B

Data Sheet Pengontrol Mikro 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
- High Endurance Non-volatile Memory segments
  - 16K Bytes of In-System Self-programmable Flash program memory
  - 512 Bytes EEPROM
  - 1K Byte Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 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 QFN/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
- Power Consumption @ 1 MHz, 3V, and 25°C for ATmega16L
  - Active: 1.1 mA
  - Idle Mode: 0.35 mA
  - Power-down Mode: < 1 µA

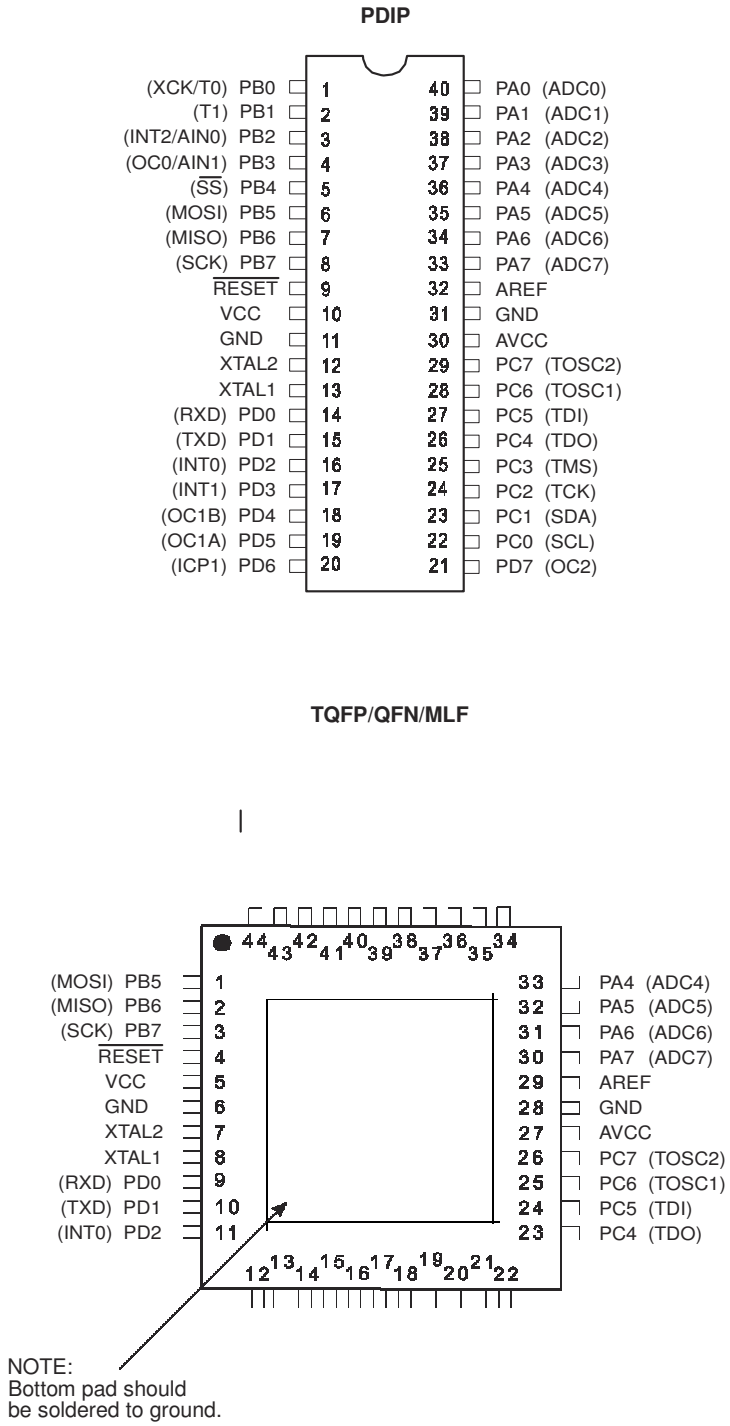
8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

ATmega16
ATmega16L

Summary

Pin Configurations

Figure 1. Pinout ATmega16



Disclaimer

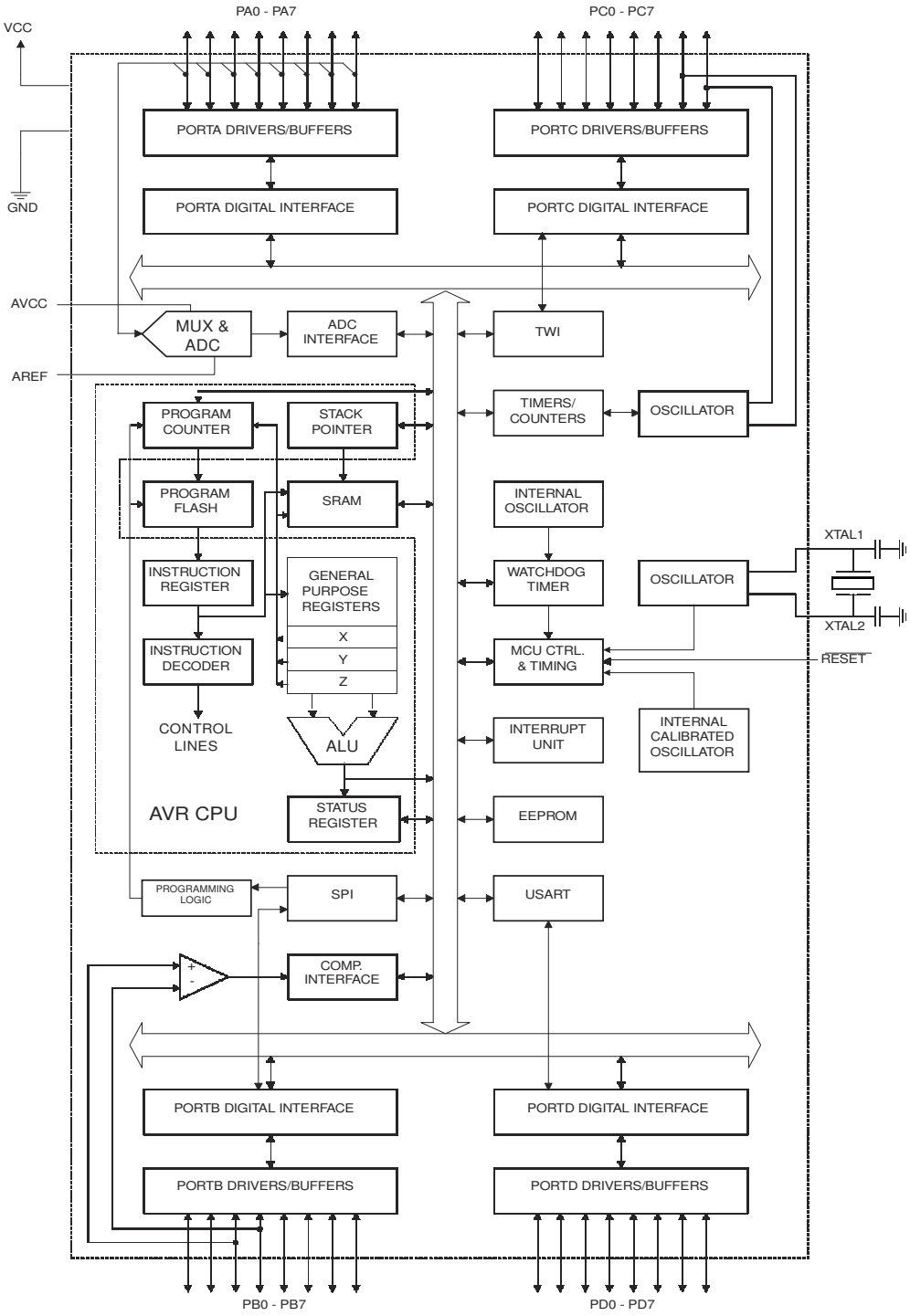
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

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.

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 58.

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 61.

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 63.

$\overline{\text{RESET}}$

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 38. 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.

Features

- Utilizes the AVR® RISC Architecture
- AVR – High-performance and Low-power RISC Architecture
  - 120 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
- Data and Non-volatile Program and Data Memories
  - 2K Bytes of In-System Self Programmable Flash
    Endurance 10,000 Write/Erase Cycles
  - 128 Bytes In-System Programmable EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 128 Bytes Internal SRAM
  - Programming Lock for Flash Program and EEPROM Data Security
- Peripheral Features
  - One 8-bit Timer/Counter with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Modes
  - Four PWM Channels
  - On-chip Analog Comparator
  - Programmable Watchdog Timer with On-chip Oscillator
  - USI – Universal Serial Interface
  - Full Duplex USART
- Special Microcontroller Features
  - debugWIRE On-chip Debugging
  - In-System Programmable via SPI Port
  - External and Internal Interrupt Sources
  - Low-power Idle, Power-down, and Standby Modes
  - Enhanced Power-on Reset Circuit
  - Programmable Brown-out Detection Circuit
  - Internal Calibrated Oscillator
- I/O and Packages
  - 18 Programmable I/O Lines
  - 20-pin PDIP, 20-pin SOIC, 20-pad QFN/MLF
- Operating Voltages
  - 1.8 - 5.5V (ATtiny2313V)
  - 2.7 - 5.5V (ATtiny2313)
- Speed Grades
  - ATtiny2313V: 0 - 4 MHz @ 1.8 - 5.5V, 0 - 10 MHz @ 2.7 - 5.5V
  - ATtiny2313: 0 - 10 MHz @ 2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V
- Typical Power Consumption
  - Active Mode
    1 MHz, 1.8V: 230 µA
    32 kHz, 1.8V: 20 µA (including oscillator)
  - Power-down Mode
    < 0.1 µA at 1.8V

8-bit AVR® Microcontroller with 2K Bytes In-System Programmable Flash

ATtiny2313/V

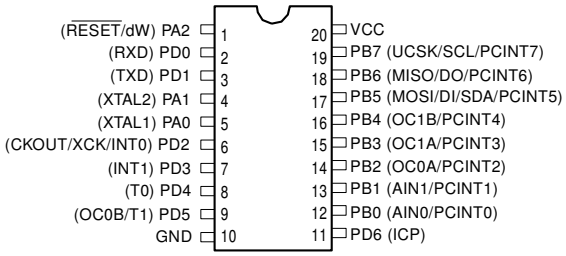
Preliminary Summary

Rev. 2543IS–AVR–04/06

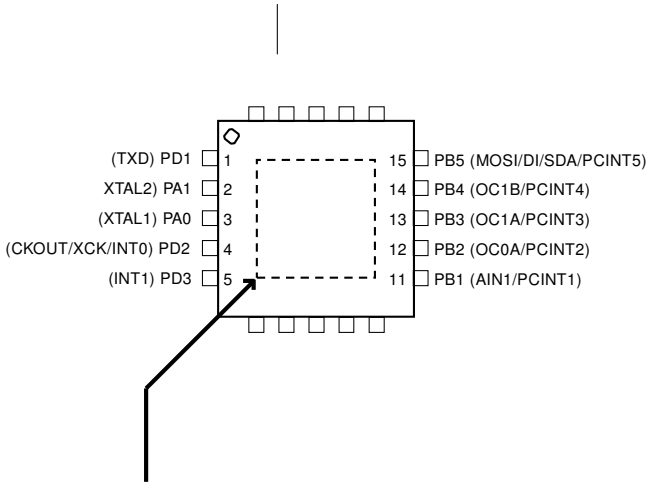
Pin Configurations

Figure 1. Pinout ATtiny2313

PDIP/SOIC



MLF



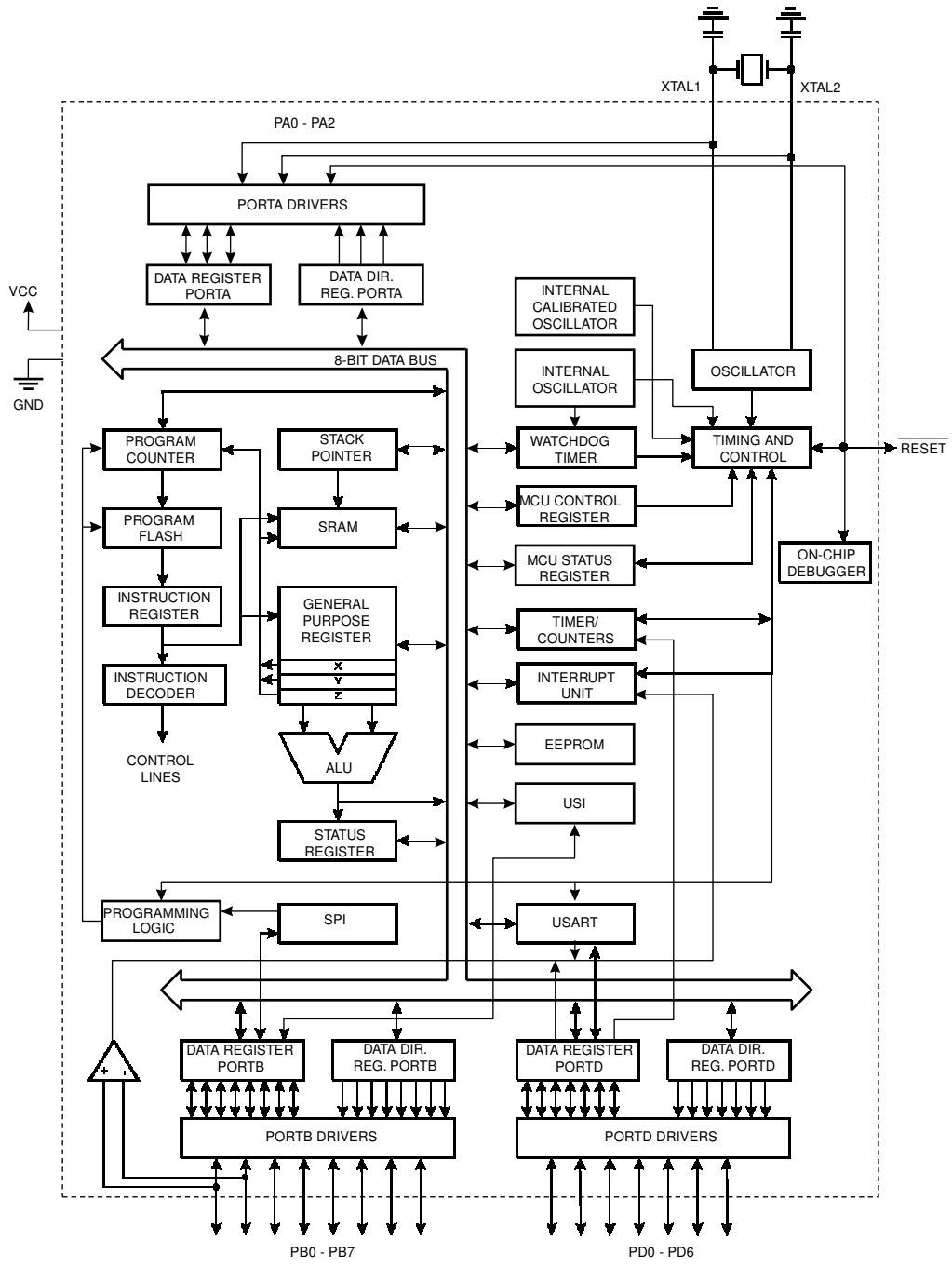
NOTE: Bottom pad should be soldered to ground.

Overview

The ATtiny2313 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 ATtiny2313 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 ATtiny2313 provides the following features: 2K bytes of In-System Programmable Flash, 128 bytes EEPROM, 128 bytes SRAM, 18 general purpose I/O lines, 32 general purpose working registers, a single-wire Interface for On-chip Debugging, two flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, Universal Serial Interface with Start Condition Detector, a programmable Watchdog Timer with internal Oscillator, and three software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, 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 interrupt or hardware reset. 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.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, or by a conventional non-volatile memory programmer. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATtiny2313 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATtiny2313 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 (PA2..PA0) Port A is a 3-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the 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.

Port A also serves the functions of various special features of the ATtiny2313 as listed on page 53.

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 ATtiny2313 as listed on page 53.

Port D (PD6..PD0) Port D is a 7-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 ATtiny2313 as listed on page 56.

$\overline{\text{RESET}}$ 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 34. Shorter pulses are not guaranteed to generate a reset. The Reset Input is an alternate function for PA2 and dW.

XTAL1 Input to the inverting Oscillator amplifier and input to the internal clock operating circuit. XTAL1 is an alternate function for PA0.

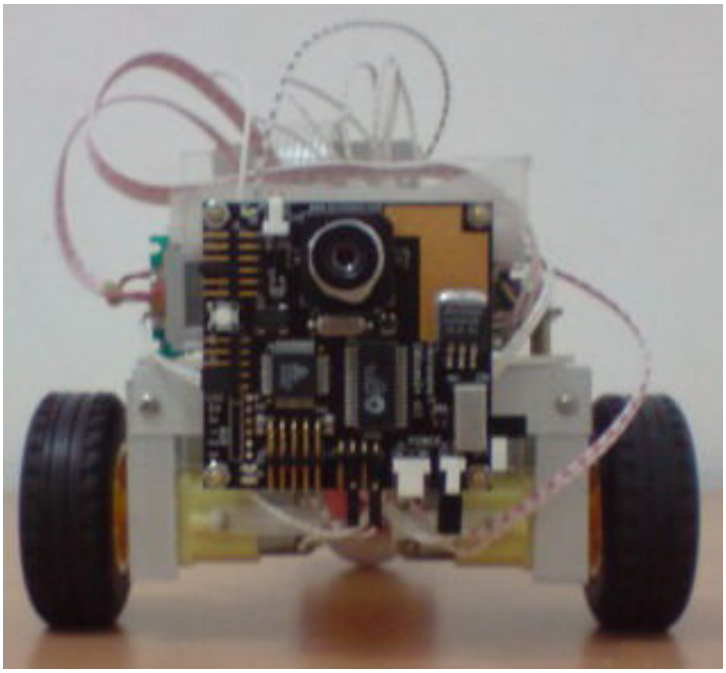
XTAL2 Output from the inverting Oscillator amplifier. XTAL2 is an alternate function for PA1.

Resources

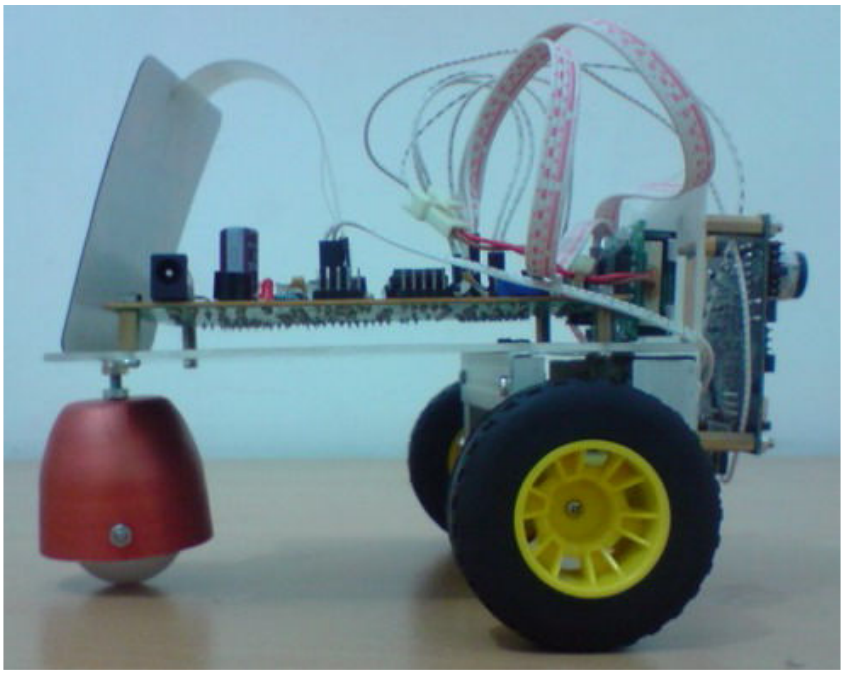
A comprehensive set of development tools, application notes and datasheets are available for downloadon http://www.atmel.com/avr.

LAMPIRAN - C

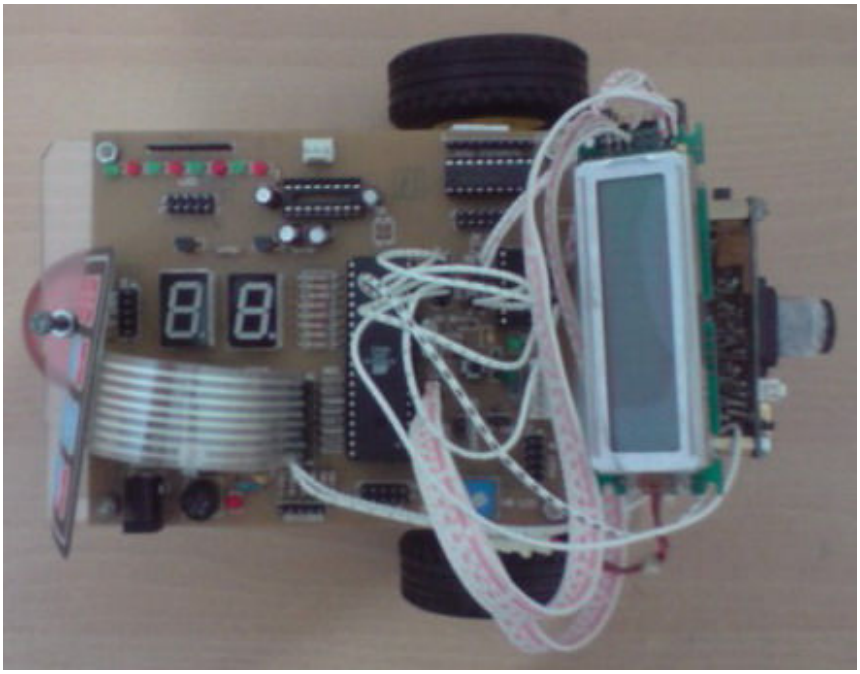
Foto Alat



Tampak depan



Tampak samping



Tampak atas

Tampak belakang