Features

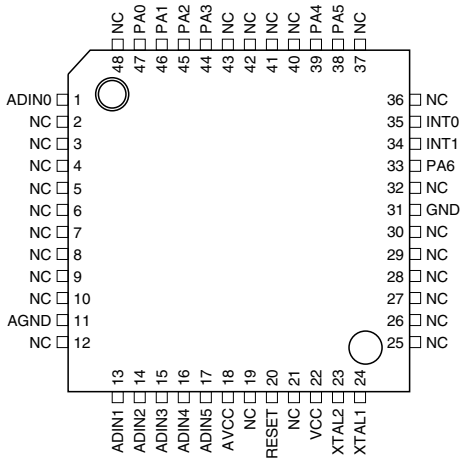
- Utilizes the AVR® RISC Architecture
- AVR – High-performance and Low-power RISC Architecture
  - 118 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General-purpose Working Registers
  - Up to 1.5 MIPS Throughput at 1.5 MHz
- Data and Nonvolatile Program Memory
  - 8K Bytes Flash Program Memory
    Endurance: 1,000 Write/Erase Cycles
  - 256 Bytes Internal SRAM
  - 512 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - Programming Lock for Flash Program and EEPROM Data Security
- Peripheral Features
  - One 8-bit Timer/Counter with Separate Prescaler
  - One 16-bit Timer/Counter with Separate Prescaler
- Special Microcontroller Features
  - Low-power Idle and Power-down Modes
  - External and Internal Interrupt Sources
  - 6-channel, 10-bit ADC
- Specifications
  - Low-power, High-speed CMOS Process Technology
  - Fully Static Operation
- Power Consumption at 1.5 MHz, 3.6V, 25°C
  - Active: 1.2 mA
  - Idle Mode: 0.2 mA
  - Power-down Mode: <10 µA
- I/O and Packages
  - Seven General Output Lines
  - Two External Interrupt Lines
  - 48-lead LQFP/VQFP Package
- Operating Voltage
  - 3.3 - 6.0V
- Speed Grade
  - 0 - 1.5 MHz

Description

The AT90C8534 is a low-power CMOS 8-bit microcontroller based on the AVR RISC architecture. By executing powerful instructions in a single clock cycle, the

(continued)

Pin Configuration



8-bit AVR® Microcontroller with 8K Bytes Programmable Flash

AT90C8534

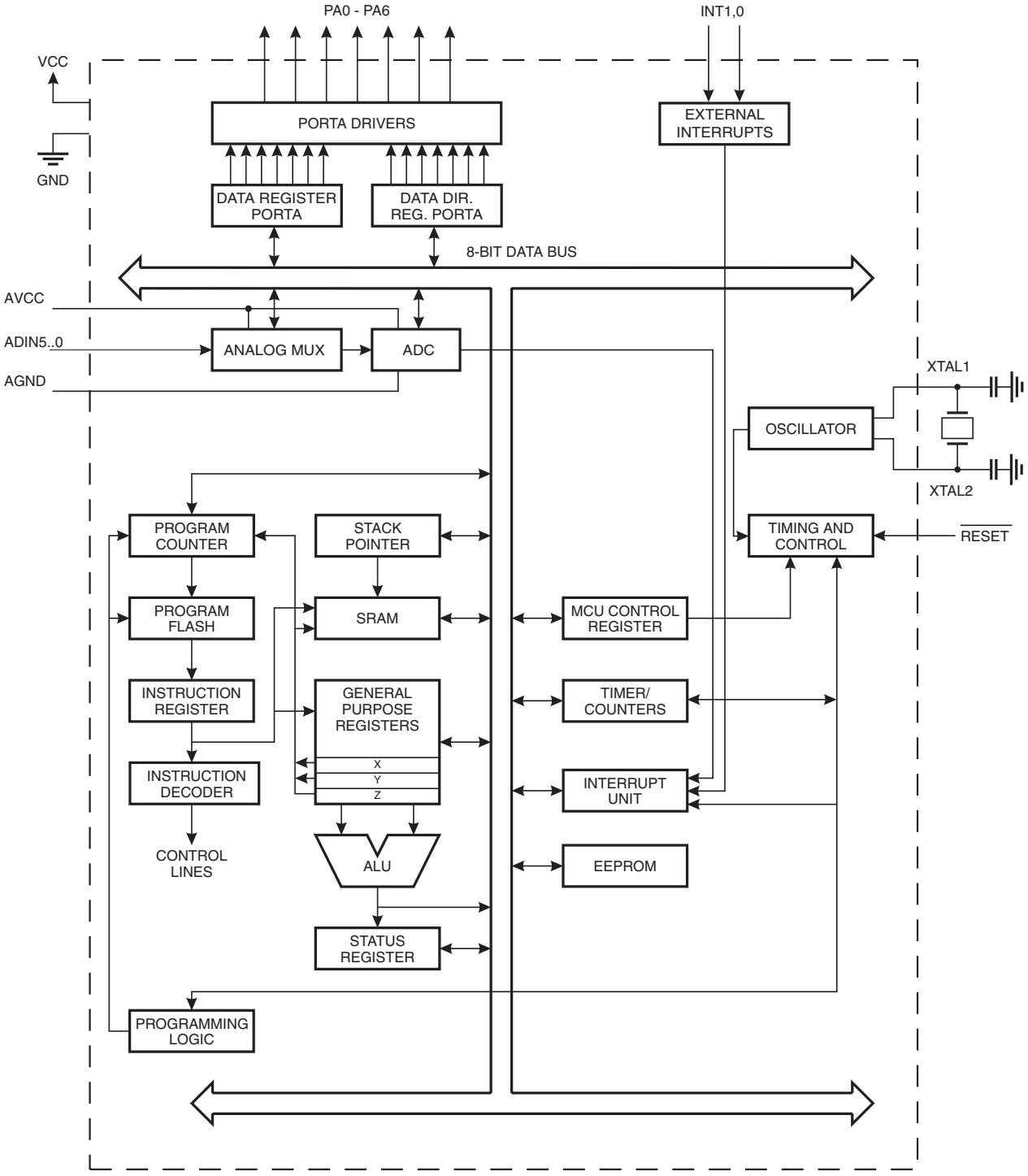
Preliminary

Rev. 1229B–11/00

AT90C8534 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 1. The AT90C8534 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 AT90C8534 provides the following features: 8K bytes of programmable Flash, 512 bytes EEPROM, 256 bytes SRAM, 7 general output lines, 2 external interrupt lines, 32 general-purpose working registers, 2 flexible timer/counters, internal and external interrupts, 6-channel, 10-bit ADC, and 2 software-selectable power saving modes. The Idle mode stops the CPU while allowing the ADC, timer/counters and interrupt system to continue functioning. The Power-down mode saves the SRAM and register contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The on-chip programmable Flash allows the program memory to be reprogrammed by a conventional nonvolatile memory programmer. By combining an 8-bit RISC CPU with programmable Flash on a monolithic chip, the Atmel AT90C8534 is a powerful microcontroller that provides a highly flexible and cost-effective solution to many embedded control applications.

The AT90C8534 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

Digital ground

Port A (PA6..PA0)

Port A is a 7-bit output port with tri-state mode. The Port A output buffers can sink 20 mA and can drive LED displays directly. The port pins are tri-stated when a reset condition becomes active, even if the clock is not running.

INT1, 0

External interrupt input pins. A falling or rising edge on either of these pins will generate an interrupt request. Interrupt pulses longer than 40 ns will generate an interrupt, even if the clock is not running.

ADIN5..0

ADC input pins. Any of these pins can be selected as the input to the ADC.

$\overline{\text{RESET}}$

Reset input. An external reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than 100 ns will generate a reset, even if the clock is not running. 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

This is the supply voltage pin for the A/D Converter. If the ADC is not used, the pin must be connected to V_{CC}. If the ADC is used, the pin should be connected to VCC via a low-pass filter. See page 30 for details on operation of the ADC.

AGND

Analog ground. If the board has a separate analog ground plane, this pin should be connected to this ground plane. Otherwise, connect to GND.

Crystal Oscillators

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or a ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 3. Note that XTAL2 should not be used to drive other components.

Figure 2. Oscillator Connections

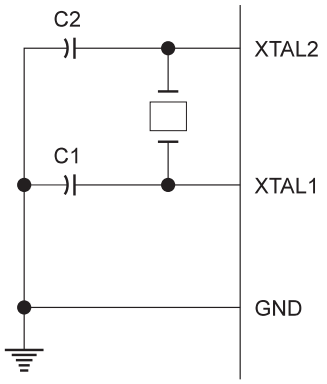
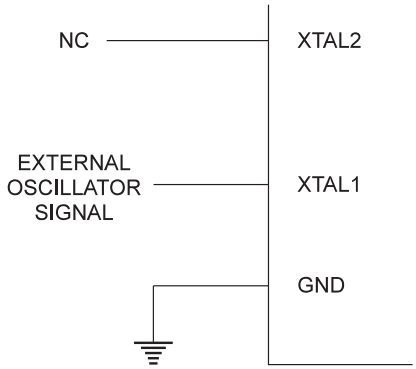


Figure 3. External Clock Drive Configuration



Architectural Overview

The fast-access register file concept contains 32 x 8-bit general-purpose working registers with a single clock cycle access time. This means that during one single clock cycle, one ALU (Arithmetic Logic Unit) operation is executed. 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 three address pointers is also used as the address pointer for the constant table look-up function. These added function registers are the 16-bit X-register, Y-register and Z-register.

The ALU supports arithmetic and logic functions between registers or between a constant and a register. Single register operations are also executed in the ALU. Figure 4 shows the AT90C8534 AVR RISC microcontroller architecture.

In addition to the register operation, the conventional memory addressing modes can be used on the register file as well. This is enabled by the fact that the register file is assigned the 32 lowermost Data Space addresses ($00 - $1F), allowing them to be accessed as though they were ordinary memory locations.

The I/O memory space contains 64 addresses for CPU peripheral functions such as Control Registers, Timer/Counters, A/D converters 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.

The AVR uses a Harvard architecture concept – with separate memories and buses for program and data. The program memory is 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 programmable Flash memory.

With the relative jump and call instructions, the whole 4K word (8K bytes) address space is directly accessed. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

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 stack pointer (SP) in the reset routine (before subroutines or interrupts are executed). The 9-bit stack pointer is read/write accessible in the I/O space.

The 256 bytes data SRAM can be easily 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.

Figure 4. The AT90C8534 AVR RISC Architecture

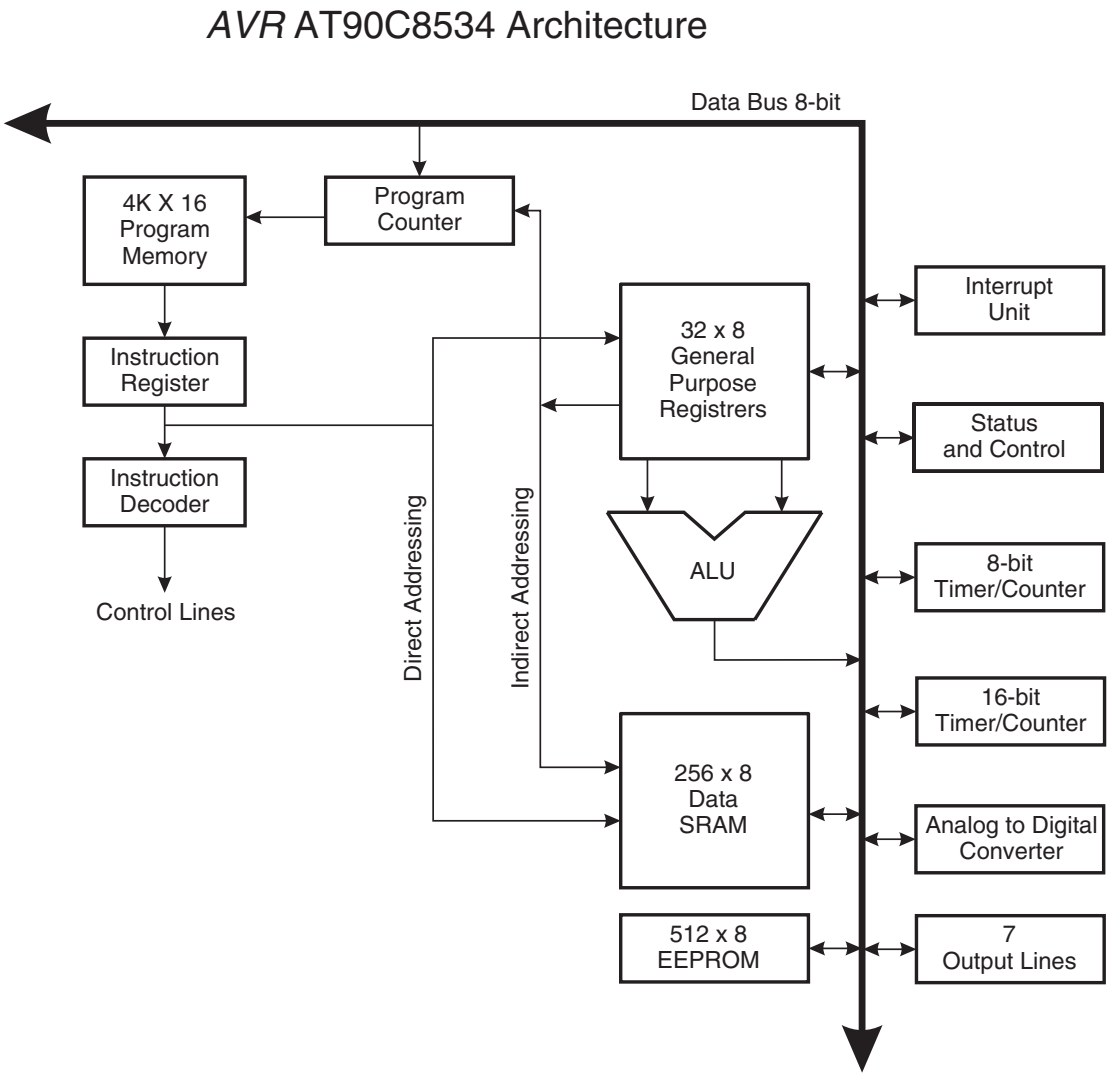
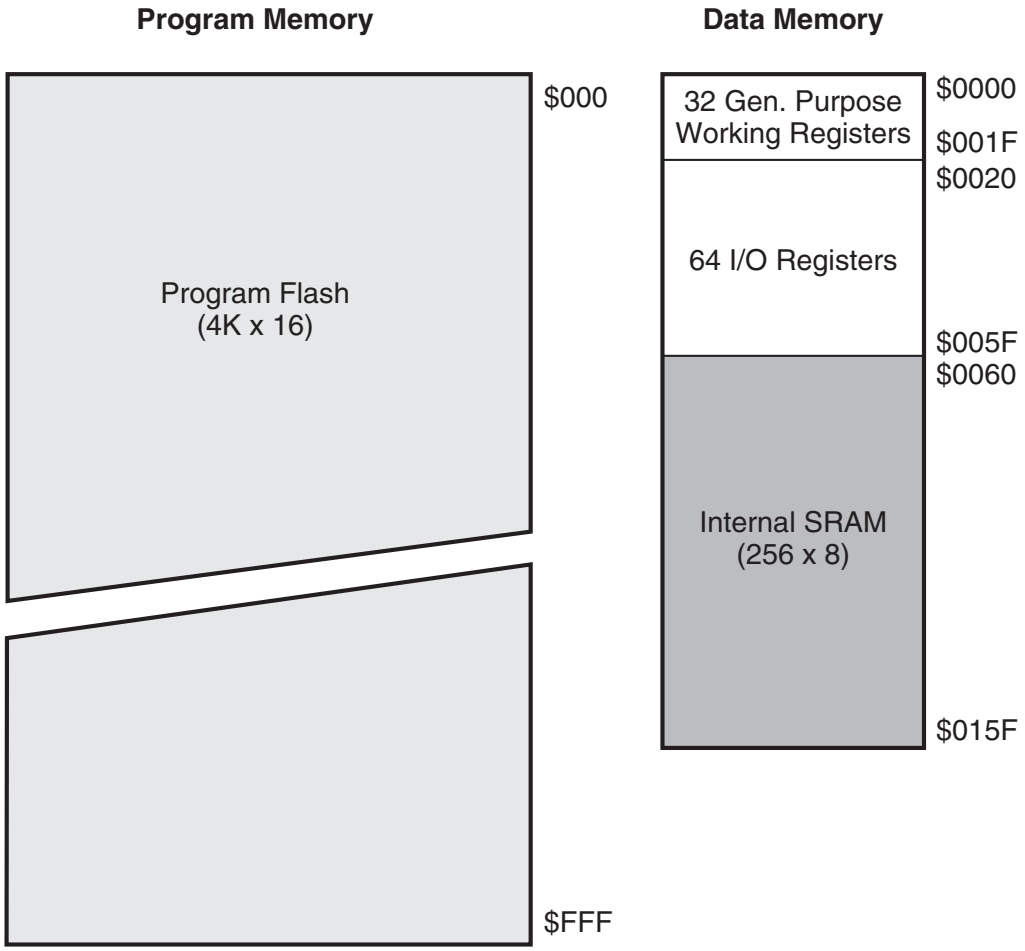


Figure 5. Memory Maps

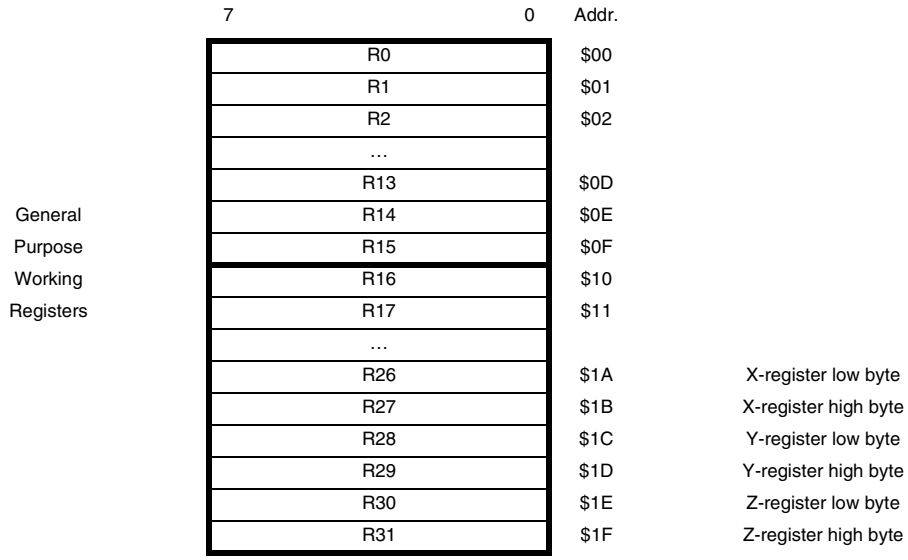


A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All the different interrupts have a separate interrupt vector in the interrupt vector table at the beginning of the program memory. The different interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

General-purpose Register File

Figure 6 shows the structure of the 32 general-purpose working registers in the CPU.

Figure 6. AVR CPU General-purpose Working Registers



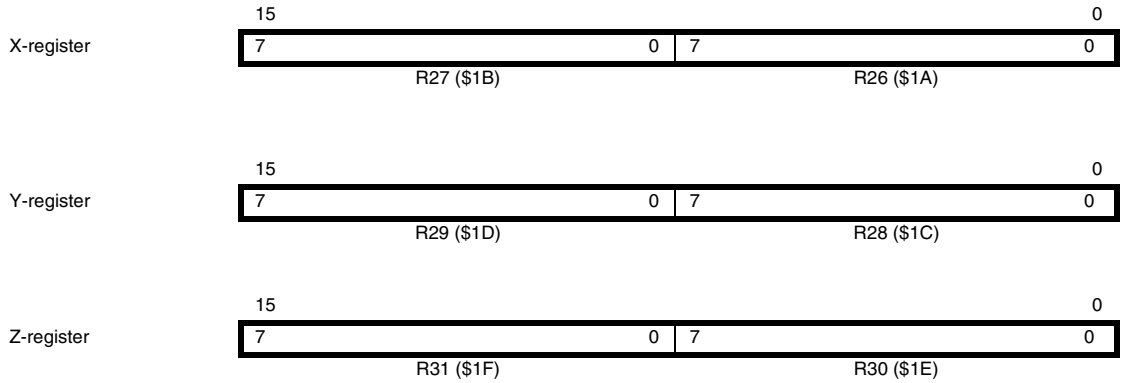
All the register operating instructions in the instruction set have direct and single-cycle access to all registers. The only exception are the five constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI and ORI between a constant and a register and the LDI instruction for load immediate constant data. These instructions apply to the second half of the registers in the register file – R16..R31. The general SBC, SUB, CP, AND and OR and all other operations between two registers or on a single register apply to the entire register file.

As shown in Figure 6, 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-registers can be set to index any register in the file.

X-register, Y-register and Z-register

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

Figure 7. X-, Y- and Z-registers



In the different addressing modes these address registers have functions as fixed displacement, automatic increment and decrement (see the descriptions for the different instructions).

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, ALU operations between registers in the register file are executed. The ALU operations are divided into three main categories: arithmetic, logical and bit functions.

Programmable Flash Program Memory

The AT90C8534 contains 8K bytes of on-chip programmable Flash memory for program storage. Since all instructions are 16- or 32-bit words, the Flash is organized as 4K x 16. The Flash memory has an endurance of at least 1000 write/erase cycles. The AT90C8534 program counter (PC) is 12 bits wide, thus addressing the 4096 program memory addresses.

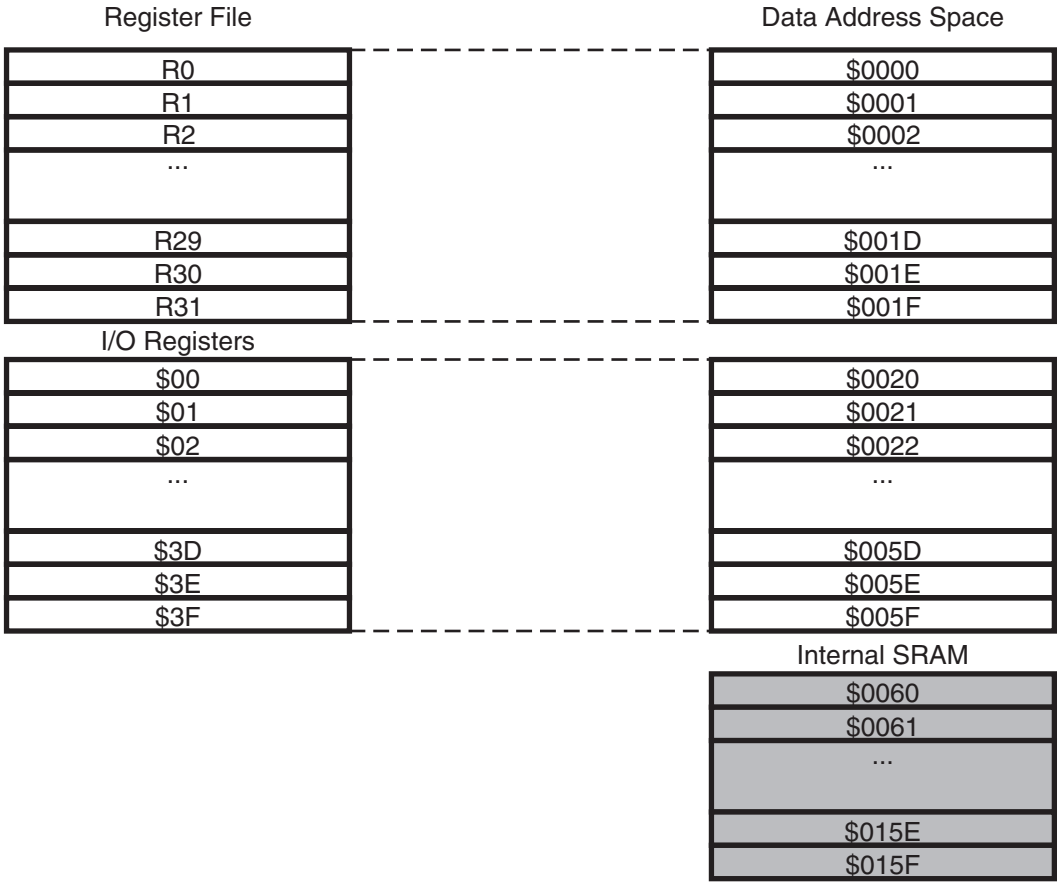
Constant tables must be allocated within the address 0 - 4K (see the LPM – Load Program Memory instruction description).

See page 9 for the different program memory addressing modes.

SRAM Data Memory

The following figure shows how the AT90C8534 SRAM memory is organized.

Figure 8. SRAM Organization



The lower 352 data memory locations address the register file, the I/O memory and the internal data SRAM. The first 96 locations address the register file + I/O memory, and the next 256 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement and Indirect with Post-increment. In the register file, registers R26 to R31 feature the indirect addressing pointer registers.

The direct addressing reaches the entire data space.

The Indirect with Displacement mode features 63 address locations reached from the base address given by the Y- or Z-register.

When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y and Z are decremented and incremented.

The 32 general-purpose working registers, 64 I/O registers and the 256 bytes of internal data SRAM in the AT90C8534 are all accessible through all these addressing modes.

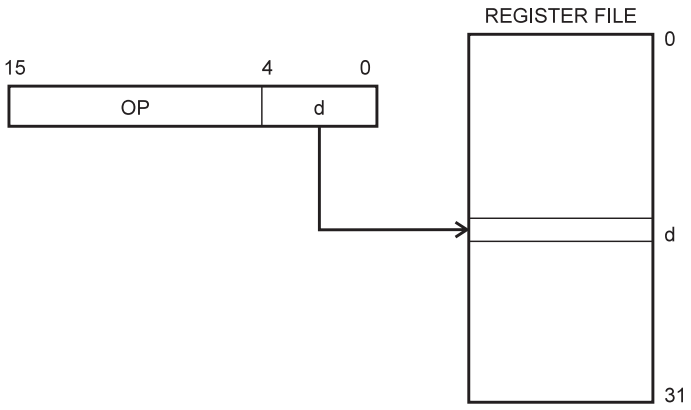
See the next section for a detailed description of the different addressing modes.

Program and Data Addressing Modes

The AT90C8534 AVR RISC microcontroller supports powerful and efficient addressing modes for access to the program memory (Flash) and data memory (SRAM, Register file and I/O memory). This section describes the different addressing modes supported by the AVR architecture. In the figures, OP means the operation code part of the instruction word. To simplify, not all figures show the exact location of the addressing bits.

Register Direct, Single Register Rd

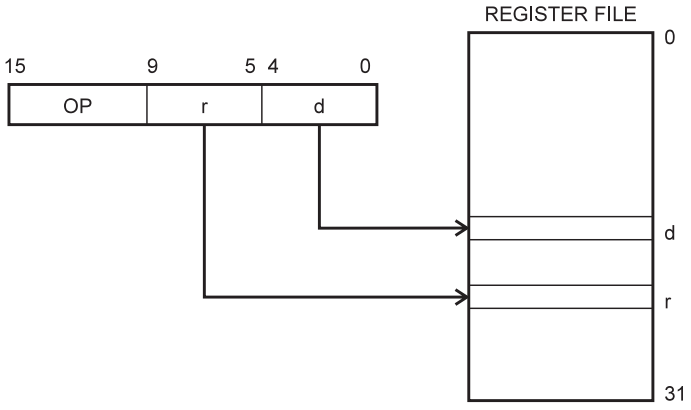
Figure 9. Direct Single Register Addressing



The operand is contained in register d (Rd).

Register Direct, Two Registers Rd And Rr

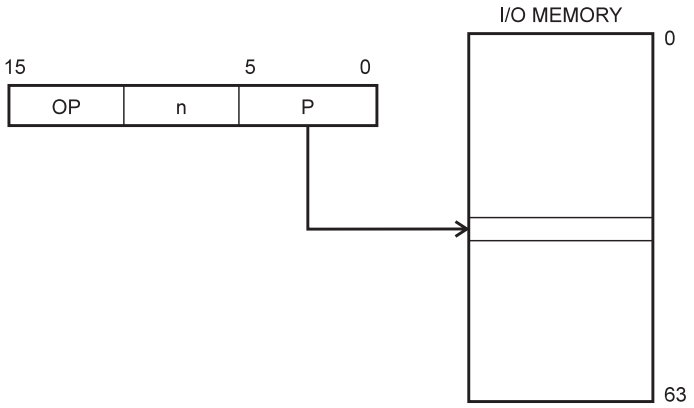
Figure 10. Direct Register Addressing, Two Registers



Operands are contained in register r (Rr) and d (Rd). The result is stored in register d (Rd).

I/O Direct

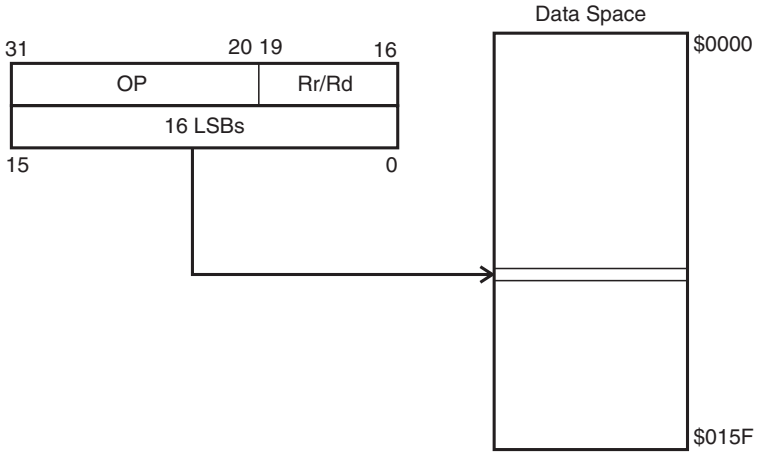
Figure 11. I/O Direct Addressing



Operand address is contained in six bits of the instruction word. n is the destination or source register address.

Data Direct

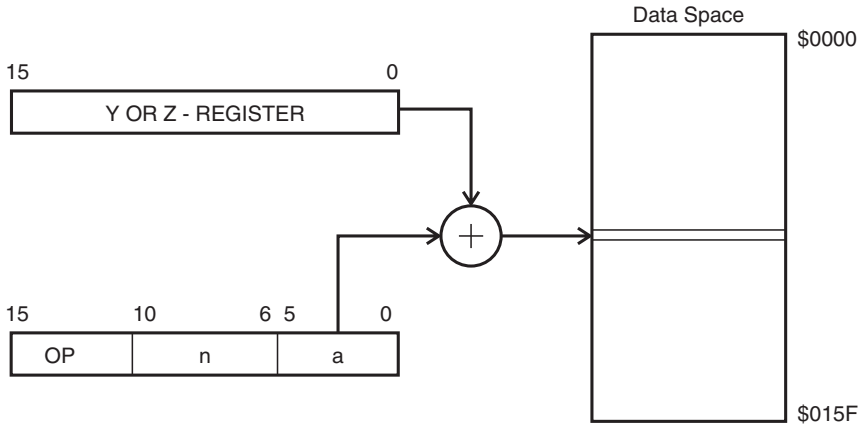
Figure 12. Direct Data Addressing



A 16-bit data address is contained in the 16 LSBs of a 2-word instruction. Rd/Rr specify the destination or source register.

Data Indirect with Displacement

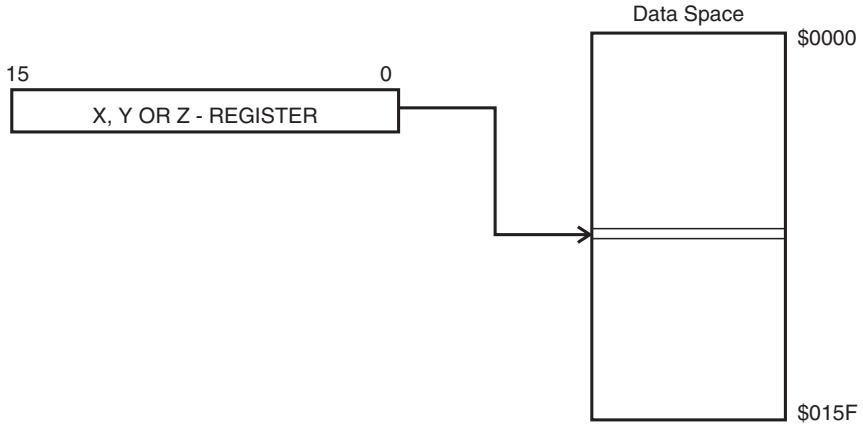
Figure 13. Data Indirect with Displacement



Operand address is the result of the Y- or Z-register contents added to the address contained in six bits of the instruction word.

Data Indirect

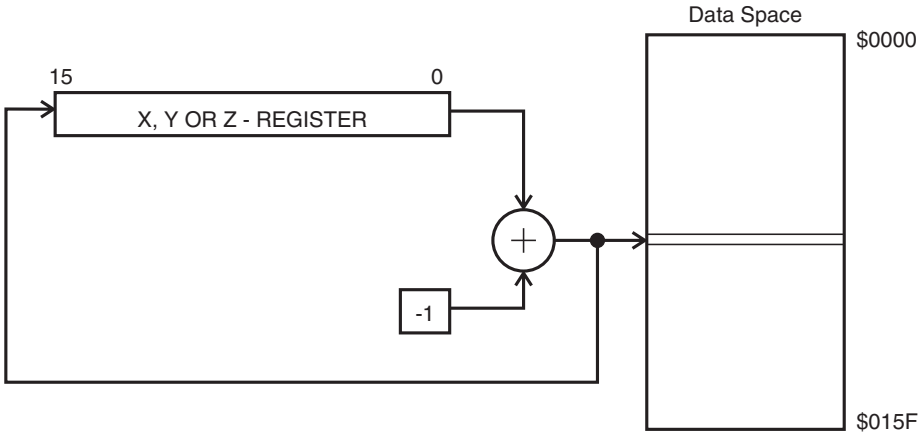
Figure 14. Data Indirect Addressing



Operand address is the contents of the X-, Y- or the Z-register.

Data Indirect with Pre-decrement

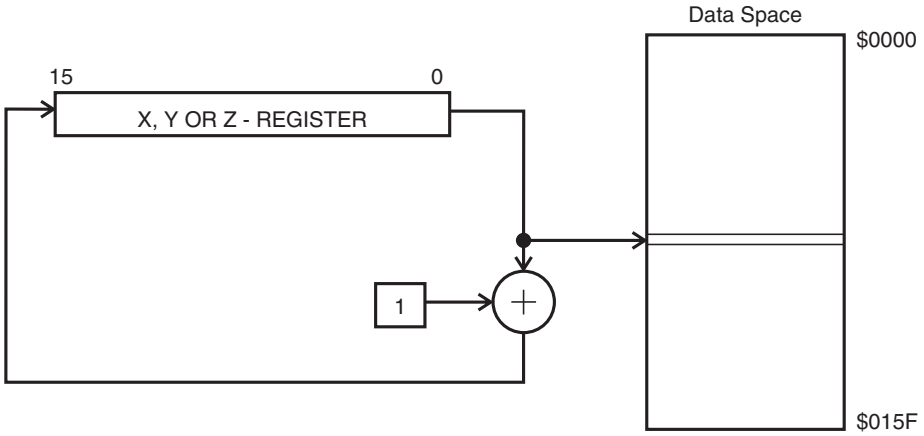
Figure 15. Data Indirect Addressing with Pre-decrement



The X-, Y- or the Z-register is decremented before the operation. Operand address is the decremented contents of the X-, Y- or the Z-register.

Data Indirect with Post-increment

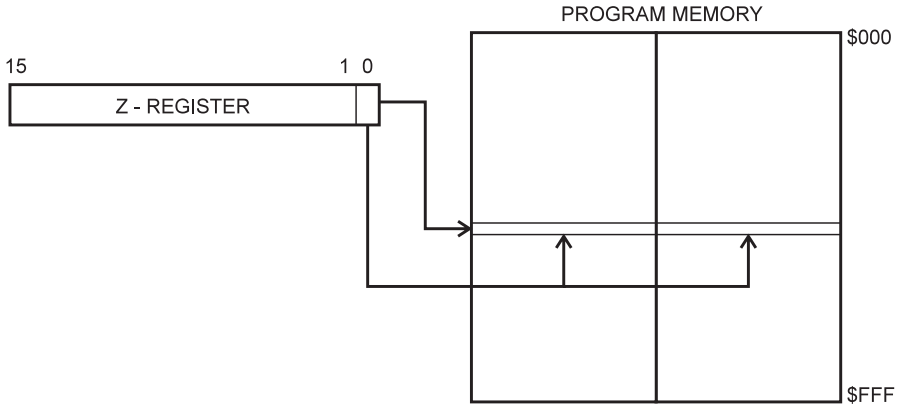
Figure 16. Data Indirect Addressing with Post-increment



The X-, Y- or the Z-register is incremented after the operation. Operand address is the content of the X-, Y- or the Z-register prior to incrementing.

Constant Addressing Using the LPM Instruction

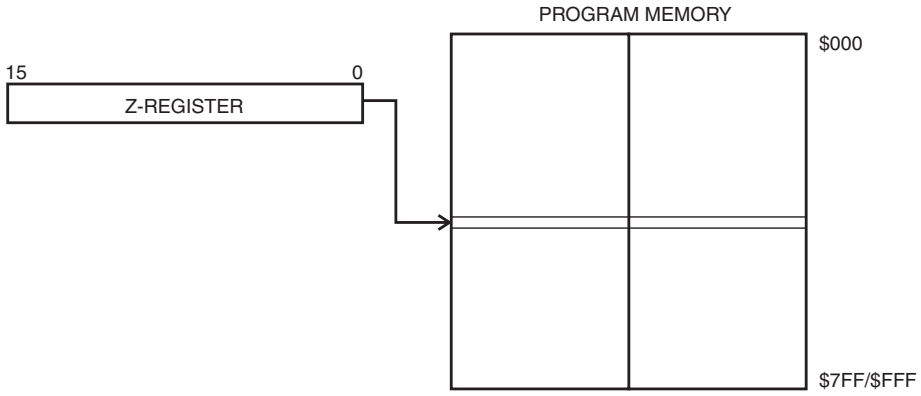
Figure 17. Code Memory Constant Addressing



Constant byte address is specified by the Z-register contents. The 15 MSBs select word address (0 - 4K), the LSB selects low byte if cleared (LSB = 0) or high byte if set (LSB = 1).

Indirect Program Addressing, IJMP and ICALL

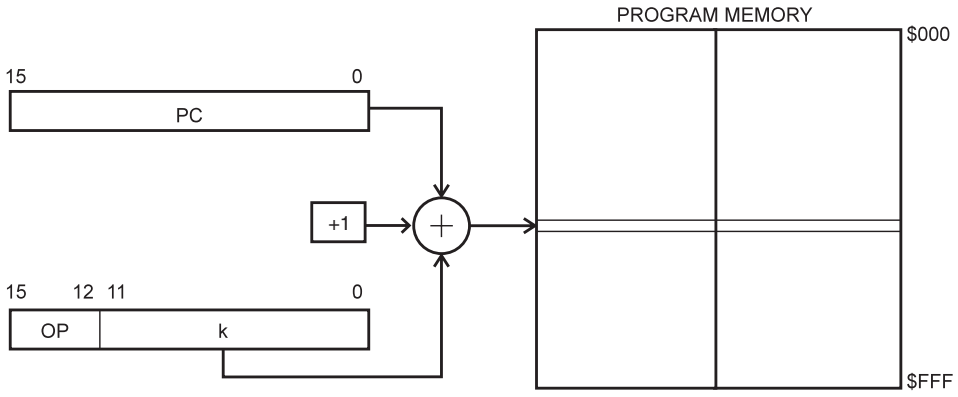
Figure 18. Indirect Program Memory Addressing



Program execution continues at address contained by the Z-register (i.e., the PC is loaded with the contents of the Z-register).

Relative Program Addressing, RJMP And RCALL

Figure 19. Relative Program Memory Addressing



Program execution continues at address PC + k + 1. The relative address k is from -2048 to 2047.

EEPROM Data Memory

The AT90C8534 contains 512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described on page 28, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

Memory Access Times and Instruction Execution Timing

This section describes the general access timing concepts for instruction execution and internal memory access.

The AVR CPU is driven by the System Clock Ø, directly generated from the external clock crystal for the chip. No internal clock division is used.

Figure 20 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks and functions per power unit.

Figure 20. The Parallel Instruction Fetches and Instruction Executions

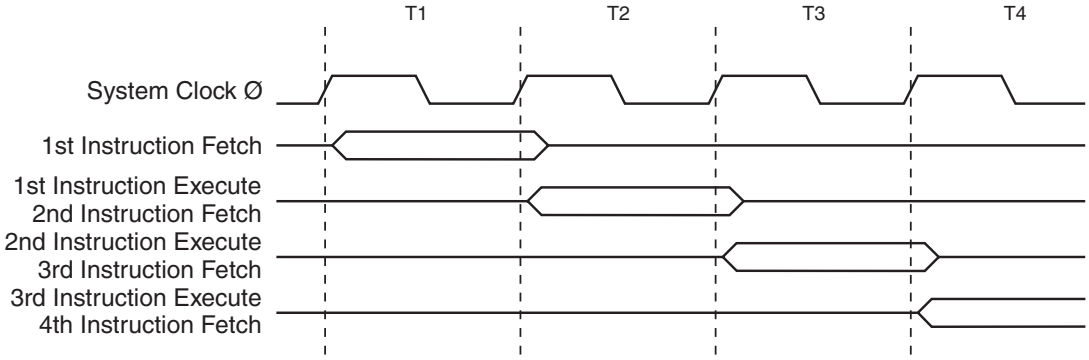
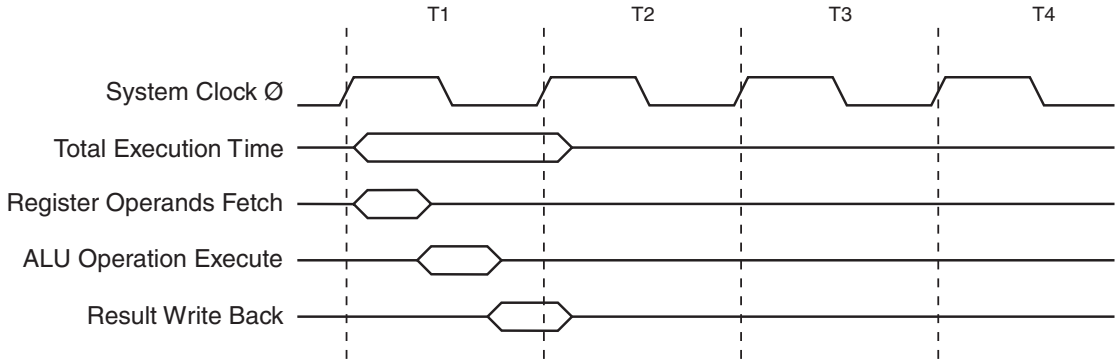


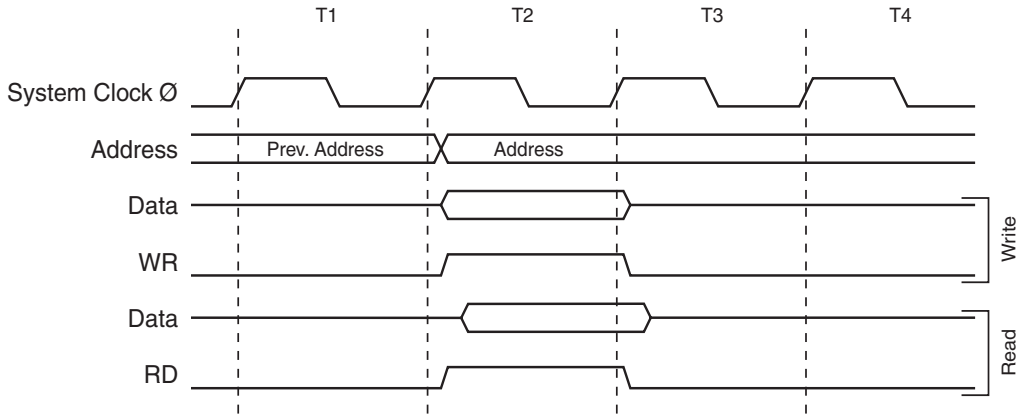
Figure 21 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed and the result is stored back to the destination register.

Figure 21. Single Cycle ALU Operation



The internal data SRAM access is performed in two System Clock cycles as described in Figure 22.

Figure 22. On-Chip Data SRAM Access Cycles



I/O Memory

The I/O space definition of the AT90C8534 is shown in Table 1.

Table 1. AT90C8534 I/O Space

I/O Address (SRAM Address)	Name	Function
$3F ($5F)	SREG	Status REGister
$3E ($5E)	SPH	Stack Pointer High
$3D ($5D)	SPL	Stack Pointer Low
$3B ($5B)	GIMSK	General Interrupt MaSK register
$3A ($5A)	GIFR	General Interrupt Flag Register
$39 ($59)	TIMSK	Timer/Counter Interrupt MaSK register
$38 ($58)	TIFR	Timer/Counter Interrupt Flag register
$35 ($55)	MCUCR	MCU general Control Register
$33 ($53)	TCCR0	Timer/Counter0 Control Register
$32 ($52)	TCNT0	Timer/Counter0 (8-bit)
$2E ($4E)	TCCR1	Timer/Counter1 Control Register
$2D ($4D)	TCNT1H	Timer/Counter1 High Byte
$2C ($4C)	TCNT1L	Timer/Counter1 Low Byte
$1F ($3E)	EEARH	EEPROM Address Register High Byte
$1E ($3E)	EEARL	EEPROM Address Register Low Byte
$1D ($3D)	EEDR	EEPROM Data Register
$1C ($3C)	EECR	EEPROM Control Register
$1B ($3B)	PORTA	Data Register, Port A
$1A ($3A)	DDRA	Data Direction Register, Port A
$10 ($30)	GIPR	General Interrupt Pin Register
$07 ($27)	ADMUX	ADC Multiplexer Select Register
$06 ($26)	ADCSR	ADC Control and Status Register
$05 ($25)	ADCH	ADC Data Register High
$04 ($24)	ADCL	ADC Data Register Low

Note: Reserved and unused locations are not shown in the table.

The AT90C8534 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 SRAM, $20 must be added to this address. All I/O register addresses throughout this document are shown with the SRAM address in parentheses.

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

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

Status Register – SREG

The AVR status register (SREG) at I/O space location $3F ($5F) is defined as:

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	**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 (one) for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the global interrupt enable bit is cleared (zero), none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware when an interrupt routine is entered and is set by the RETI instruction to enable subsequent interrupts.

- **Bit 6 – T: Bit Copy Storage**

The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source and 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 arithmetical operations. See the Instruction Set description for detailed information.

- **Bit 4 – S: Sign Bit, S = N $\oplus$ 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 from an arithmetical or logical operations. See the Instruction Set description for detailed information.

- **Bit 1 – Z: Zero Flag**

The zero flag Z indicates a zero result from an arithmetical or logical operations. See the Instruction Set description for detailed information.

- **Bit 0 – C: Carry Flag**

The carry flag C indicates a carry in an arithmetical or logic operation. See the Instruction Set description for detailed information.

Note that the status register is not automatically stored when entering an interrupt routine or restored when returning from an interrupt routine. This must be handled by software.

Stack Pointer – SP

The AT90C8534 Stack Pointer is implemented as two 8-bit registers in the I/O space locations $3E ($5E) and $3D ($5D). As the AT90C8534 data memory has $15F locations, nine bits are used.

Bit	15	14	13	12	11	10	9	8	
$3E ($5E)	–	–	–	–	–	–	–	**SP8**	**SPH**
$3D ($5D)	**SP7**	**SP6**	**SP5**	**SP4**	**SP3**	**SP2**	**SP1**	**SP0**	**SPL**
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	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	

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 is decremented by 1 when data is pushed onto the stack with the PUSH instruction and it is

decremented by 2 when data is pushed onto the stack with subroutine RCALL and interrupt. The Stack Pointer is incremented by 1 when data is popped from the stack with the POP instruction and it is incremented by 2 when data is popped from the stack with return from subroutine RET or return from interrupt RETI.

Reset and Interrupt Handling

The AT90C8534 provides six different interrupt sources. These interrupts and the separate reset vector each have a separate program vector in the program memory space. All interrupts are assigned individual enable bits that must be set (one) together with the I-bit in the status register in order to enable the interrupt.

The lowest addresses in the program memory space are automatically defined as the Reset and Interrupt vectors. The complete list of vectors is shown in Table 2. The list also determines the priority levels of the different interrupts. The lower the address, the higher the priority level. RESET has the highest priority and next is INT0 (the External Interrupt Request 0), etc.

Table 2. Reset and Interrupt Vectors

Vector No.	Program Address	Source	Interrupt Definition
1	$000	RESET	Hardware Pin
2	$001	INT0	External Interrupt Request 0
3	$002	INT1	External Interrupt Request 1
4	$003	TIMER1 OVF	Timer/Counter1 Overflow
5	$004	TIMER0 OVF	Timer/Counter0 Overflow
6	$005	ADC	ADC Conversion Complete
7	$006	EE_RDY	EEPROM Ready

The most typical program setup for the Reset and Interrupt vector addresses are:

```
Address             Labels     Code              Comments
$000                           rjmp    RESET     ; Reset Handler
$001                           rjmp    EXT_INT0  ; IRQ0 Handler
$002                           rjmp    EXT_INT1  ; IRQ1 Handler
$003                           rjmp    TIM1_OVF  ; Timer1 Overflow Handler
$004                           rjmp    TIM0_OVF  ; Timer0 Overflow Handler
$005                           rjmp    ADC       ; ADC Conversion Complete Interrupt Handler
$006                           rjmp    EE_RDY    ; EEPROM Ready Handler
;
$007                MAIN:      ldi     r16, high(RAMEND); Main program start
                               out     SPH, r16
                               ldi     r16, low(RAMEND)
                               out     SPL, r16
                               <instr>  xxx
…                   …            …       …
```

Reset

During reset, all I/O registers are set to their initial values and the program counter is set to address $000. When reset is released, the program starts execution from this address. The instruction placed in address $000 must be an RJMP (relative jump) instruction to the reset handling routine. If the program never enables an interrupt source, the interrupt vectors are not used and regular program code can be placed at these locations. The circuit diagram in Figure 23 shows the reset logic. Table 3 defines the timing and electrical parameters of the reset circuitry

Figure 23. Reset Logic

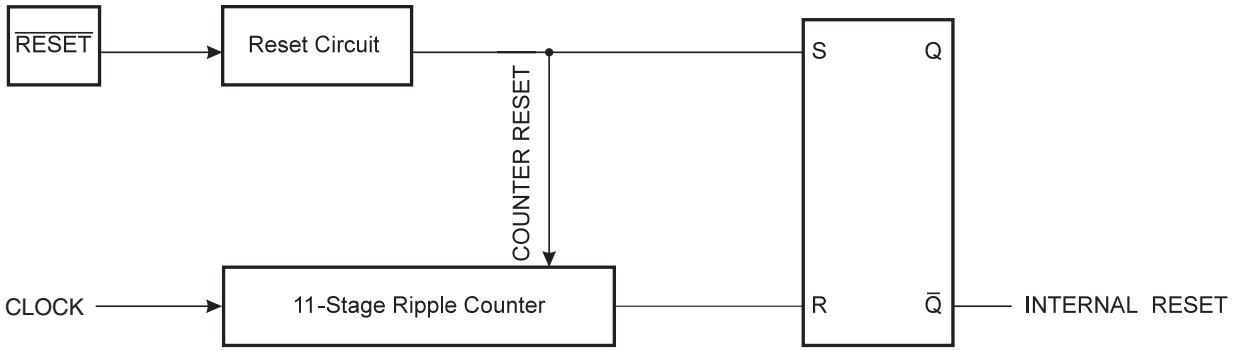


Table 3. Reset Characteristics (V_{CC} = 5.0V)

Symbol	Parameter	Min	Typ	Max	Units
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage		0.6 V_{CC}		V
t_{TOUT}	Reset Delay Time-out Period	-	1026	-	clocks

External reset

The AT90C8534 has one source of reset: the external reset pin. The external reset is used for three purposes:

1. Power-on Reset. During power-on, the external reset must be held active (low) until 100 ns after V_{CC} has reached the minimum operation voltage.
2. Brown-out Reset. If V_{CC} drops below the minimum operation voltage during operation, the external reset must go active immediately, and must be held active until 100 ns after V_{CC} rises to the minimum operation voltage.
3. Normal Operation Reset. During normal operation, reset is generated by holding the external reset active for at least 100 ns.

When the external reset is released, an internal timer that is clocked from the external clock input is started, holding the internal reset active until the external clock source has toggled a certain number of times (see Table 3). This is illustrated in Figure 24 and Figure 25.

Figure 24. External Reset on Start-up

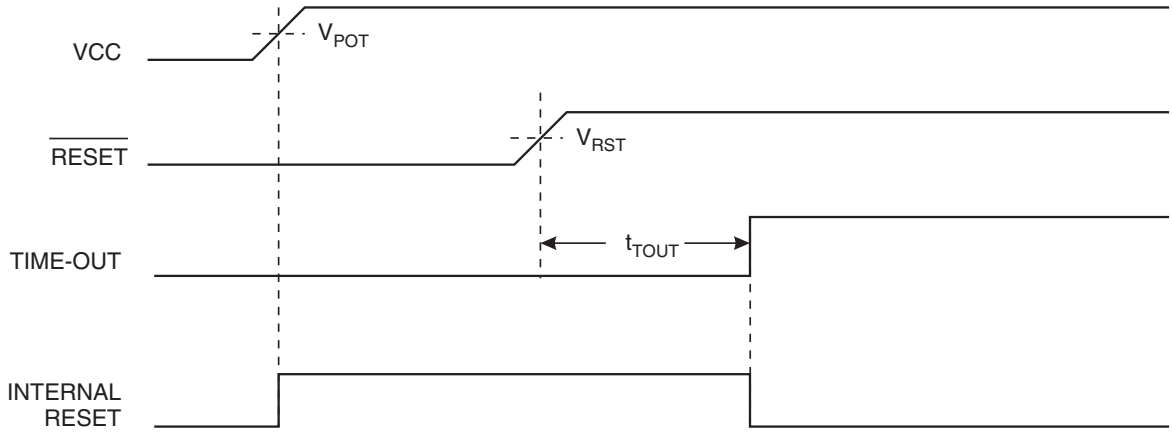
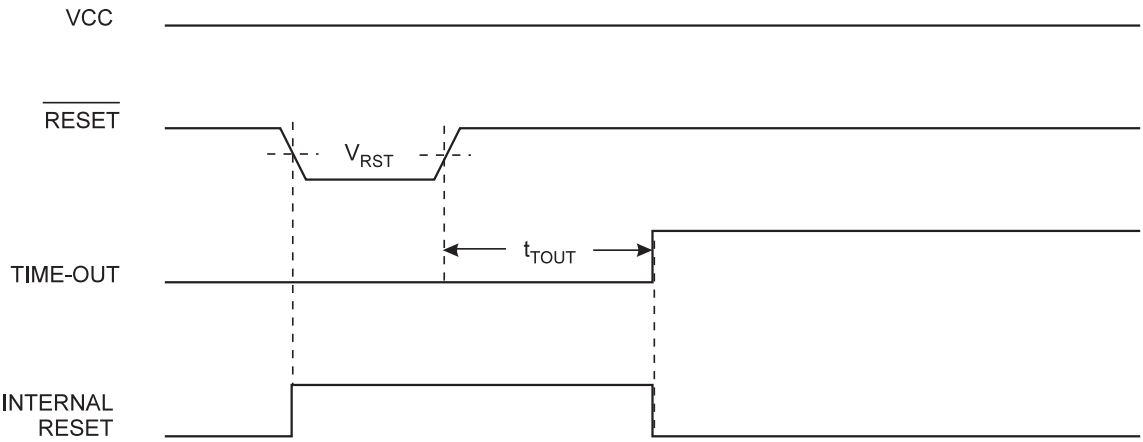


Figure 25. External Reset during Operation



Interrupt Handling

The AT90C8534 has two 8-bit Interrupt Mask control registers; GIMSK (General Interrupt Mask register) and TIMSK (Timer/Counter Interrupt Mask register).

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared (zero) and all interrupts are disabled. The user software can set (one) the I-bit to enable nested interrupts. The I-bit is set (one) when a Return from Interrupt instruction (RETI) is executed.

When the Program Counter is vectored to the actual interrupt vector in order to execute the interrupt handling routine, hardware clears the corresponding flag that generated the interrupt. Some of the interrupt flags can also be cleared by writing a logical “1” to the flag bit position(s) to be cleared.

If an interrupt condition occurs when the corresponding interrupt enable bit is cleared (zero), the interrupt flag will be set and remembered until the interrupt is enabled or the flag is cleared by software.

If one or more interrupt conditions occur when the global interrupt enable bit is cleared (zero), the corresponding interrupt flag(s) will be set and remembered until the global interrupt enable bit is set (one), and will be executed by order of priority.

Note that the status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt routine. This must be handled by software.

General Interrupt Mask Register – GIMSK

Bit	7	6	5	4	3	2	1	0	
$3B ($5B)	INT1	INT0	–	–	–	–	–	–	GIMSK
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 – INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The external interrupt is activated on falling or rising edge of the INT1 pin. The corresponding interrupt of External Interrupt Request 1 is executed from program memory address $002. See also “External Interrupts”.

- **Bit 6 – INT0: External Interrupt Request 0 Enable**

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The external interrupt is activated on falling or rising edge of the INT0 pin. The corresponding interrupt of External Interrupt Request 0 is executed from program memory address $001. See also “External Interrupts”.

- **Bits 5..0 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

General Interrupt Flag Register – GIFR

Bit	7	6	5	4	3	2	1	0	
$3A ($5A)	INTF1	INTF0	–	–	–	–	–	–	GIFR
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 – INTF1: External Interrupt Flag 1**

When an event on the INT1 pin triggers an interrupt request, INTF1 becomes set (one). If the I-bit in SREG and the INT1 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $002. The flag is cleared when fetching the interrupt vector. Alternatively, the flag can be cleared by writing a logical “1” to it.

- **Bit 6 – INTF0: External Interrupt Flag 0**

When an event on the INT0 pin triggers an interrupt request, INTF0 becomes set (one). If the I-bit in SREG and the INT0 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $001. The flag is cleared when fetching the interrupt vector. Alternatively, the flag can be cleared by writing a logical “1” to it.

- **Bits 5..0 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

General Interrupt Pin Register – GIPR

Bit	7	6	5	4	3	2	1	0	
$10 ($30)	–	–	–	–	IPIN1	IPIN0	–	–	GIPR
Read/Write	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	x	x	0	0	

- **Bits 7..4 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

- **Bit 3 – IPIN1: External Interrupt Pin 1**

Reading this bit returns the logical value present on input pin INT1 (after synchronization latches).

- **Bit 2 – IPIN0: External Interrupt Pin 0**

Reading this bit returns the logical value present on input pin INT0 (after synchronization latches).

- **Bits 1..0 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

Timer/Counter Interrupt Mask Register – TIMSK

Bit	7	6	5	4	3	2	1	0	
$39 ($59)	–	–	–	–	–	TOIE1	–	TOIE0	TIMSK
Read/Write	R	R	R	R	R	R/W	R	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

- **Bit 2 – TOIE1: Timer/Counter1 Overflow Interrupt Enable**

When the TOIE1 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 Overflow interrupt is enabled. The corresponding interrupt (at vector $003) is executed if an overflow in Timer/Counter1 occurs, i.e., when the Overflow Flag (Timer/Counter1) is set (one) in the Timer/Counter Interrupt Flag Register (TIFR).

- **Bit 1 – Res: Reserved Bit**

This bit is a reserved bit in the AT90C8534 and always reads as zero.

- **Bit 0 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt (at vector $004) is executed if an overflow in Timer/Counter0 occurs, i.e., when the Overflow Flag (Timer0) is set (one) in the Timer/Counter Interrupt Flag Register (TIFR).

Timer/Counter Interrupt Flag Register – TIFR

Bit	7	6	5	4	3	2	1	0	
$38 ($58)	–	–	–	–	–	TOV1	–	TOV0	TIFR
Read/Write	R	R	R	R	R	R/W	R	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

- **Bit 2 – TOV1: Timer/Counter1 Overflow Flag**

The TOV1 is set (one) when an overflow occurs in Timer/Counter1. TOV1 is cleared when fetching the interrupt vector. Alternatively, TOV1 is cleared by writing a logical “1” to the flag. When the I-bit in SREG and TOIE1 (Timer/Counter1 Overflow Interrupt Enable) and TOV1 are set (one), the Timer/Counter1 Overflow Interrupt is executed.

- **Bit 1 – Res: Reserved Bit**

This bit is a reserved bit in the AT90C8534 and always reads as zero.

- **Bit 0 – TOV0: Timer/Counter0 Overflow Flag**

The bit TOV0 is set (one) when an overflow occurs in Timer/Counter0. TOV0 is cleared when fetching the interrupt vector. Alternatively, TOV0 is cleared by writing a logical “1” to the flag. When the SREG I-bit and TOIE0 (Timer/Counter0 Overflow Interrupt Enable) and TOV0 are set (one), the Timer/Counter0 Overflow interrupt is executed.

External Interrupts

The external interrupts are triggered by the INT1 and INT0 pins. The external interrupts can be triggered by a falling or rising edge.

Interrupt Response Time

The interrupt execution response for all the enabled *AVR* interrupts is four clock cycles minimum. Four clock cycles after the interrupt flag has been set, the program vector address for the actual interrupt handling routine is executed. During this 4-clock-cycle period, the Program Counter (2 bytes) is pushed onto the stack and the Stack Pointer is decremented by 2. The vector is normally a relative jump to the interrupt routine, and this jump takes two clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served.

A return from an interrupt handling routine (same as for a subroutine call routine) takes four clock cycles. During these four clock cycles, the Program Counter (2 bytes) is popped back from the stack, the Stack Pointer is incremented by 2 and the I-flag in SREG is set. When the AVR exits from an interrupt, it will always return to the main program and execute one more instruction before any pending interrupt is served.

MCU Control Register – MCUCR

The MCU Control Register contains control bits for general MCU functions.

Bit	7	6	5	4	3	2	1	0	
$35 ($55)	–	SE	SM	–	–	ISC1	–	ISC0	MCUCR
Read/Write	R	R/W	R/W	R	R	R/W	R	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: Reserved Bit**

This bit is reserved bits in the AT90C8534 and always reads as zero.

- **Bit 6 – SE: Sleep Enable**

The SE bit must be set (one) to make the MCU enter the sleep mode when the SLEEP instruction is executed. To avoid the MCU entering the sleep mode, unless it is the programmer's purpose, it is recommended to set the Sleep Enable (SE) bit just before the execution of the SLEEP instruction.

- **Bit 5 – SM: Sleep Mode**

This bit selects between the two available sleep modes. When SM is cleared (zero), Idle Mode is selected as Sleep Mode. When SM is set (one), Power-down Mode is selected as Sleep Mode. For details, refer to the section "Sleep Modes".

- **Bit 4..3 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

- **Bit 2 – ISC1: Interrupt Sense Control 1**

The external interrupt 1 is activated by the external pin INT1 if the SREG I-flag and the corresponding interrupt mask in the GIMSK are set. If ISC1 is cleared (zero) a falling edge on INT1 activates the interrupt. If ISC1 is set (one), a rising edge on INT1 activates the interrupt. Edges on INT2 are registered asynchronously. Pulses on INT1 wider than 40 ns will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt.

When changing the ISC1 bit, an interrupt can occur. Therefore, it is recommended to first disable INT1 by clearing its interrupt Enable bit in the GIMSK register. Then ISC1 bit can be changed. Finally, the INT1 interrupt flag should be cleared by writing a logical "1" to its interrupt Flag bit in the GIFR register before the interrupt is re-enabled.

- **Bit 1 – Res: Reserved Bit**

This bit is reserved bits in the AT90C8534 and always reads as zero.

- **Bit 0 – ISC0: Interrupt Sense Control 0**

The external interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask in the GIMSK are set. If ISC0 is cleared (zero), a falling edge on INT0 activates the interrupt. If ISC0 is set (one), a rising edge on INT0 activates the interrupt. Pulses on INT0 wider than 40 ns will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt.

When changing the ISC0 bit, an interrupt can occur. Therefore, it is recommended to first disable INT0 by clearing its interrupt Enable bit in the GIMSK register. Then ISC0 bit can be changed. Finally, the INT0 interrupt flag should be cleared by writing a logical "1" to its interrupt Flag bit in the GIFR register before the interrupt is re-enabled.

Sleep Modes

To enter any of the two sleep modes, the SE bit in MCUCR must be set (one) and a SLEEP instruction must be executed. The SM bit in the MCUCR register selects which sleep mode, Idle or Power-down, is activated by the SLEEP instruction.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU awakes. The CPU is then halted for four cycles, executes the interrupt routine and resumes execution from the instruction following SLEEP. The contents of the register file, SRAM and I/O memory are unaltered. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset vector.

Idle Mode

When the SM bit is cleared (zero), the SLEEP instruction forces the MCU into the Idle Mode, stopping the CPU but allowing Timer/Counters, ADC and the interrupt system to continue operating. This enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and ADC interrupts.

Power-down Mode

When the SM bit is set (one), the SLEEP instruction makes the MCU enter the Power-down Mode. In this mode, the external oscillator is stopped, while the external interrupts continue operating. Only an external reset or an external edge interrupt on INT0 or INT1 can wake up the MCU.

Note that if INT0 or INT1 is used for wake-up from Power-down Mode, the edge is remembered until the MCU wakes up.

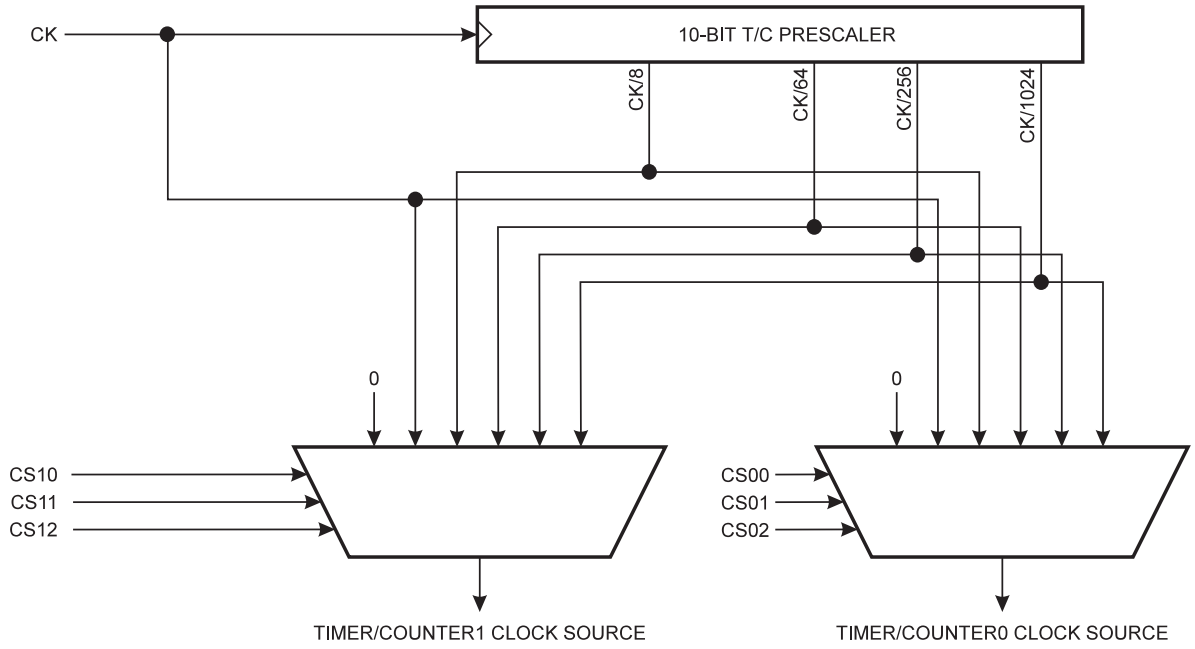
When waking up from Power-down Mode, a delay from the wake-up condition occurs until the wake-up becomes effective. This allows the clock to restart and become stable after having been stopped. The wake-up period is equal to the reset delay time-out period t_{TOUT}.

Timer/Counters

The AT90C8534 provides two general-purpose Timer/Counters – one 8-bit T/C and one 16-bit T/C. Timer/Counters 0 and 1 have individual prescaling selection from the same 10-bit prescaling timer.

Timer/Counter Prescaler

Figure 26. Prescaler for Timer/Counter0 and 1



For Timer/Counters 0 and 1, the five different prescaled selections are: CK, CK/8, CK/64, CK/256 and CK/1024, where CK is the oscillator clock. In addition, the Timer/Counters can be stopped.

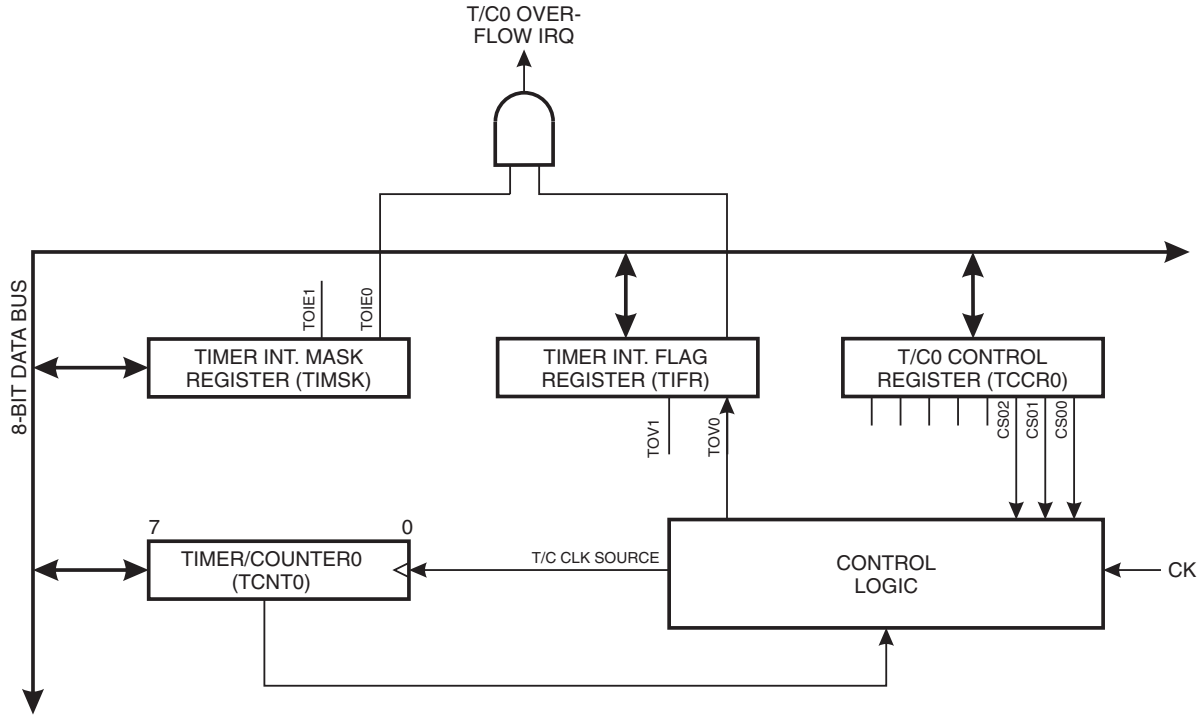
8-bit Timer/Counter0

Figure 27 shows the block diagram for Timer/Counter0.

The 8-bit Timer/Counter0 can select clock source from CK or prescaled CK. In addition, it can be stopped as described in the specification for the Timer/Counter0 Control Register – TCCR0. The overflow status flag is found in the Timer/Counter Interrupt Flag Register – TIFR. Control signals are found in the Timer/Counter0 Control Register – TCCR0. The interrupt enable/disable setting for Timer/Counter0 is found in the Timer/Counter Interrupt Mask Register – TIMSK.

The 8-bit Timer/Counter0 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make the Timer/Counter0 useful for lower speed functions or exact timing functions with infrequent actions.

Figure 27. Timer/Counter0 Block Diagram



Timer/Counter0 Control Register – TCCR0

Bit	7	6	5	4	3	2	1	0	
$33 ($53)	–	–	–	–	–	CS02	CS01	CS00	TCCR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read zero.

- **Bits 2, 1, 0 – CS02, CS01, CS00: Clock Select0, Bits 2, 1 and 0**

The Clock Select0 bits 2, 1 and 0 define the prescaling source of Timer/Counter0.

Table 4. Clock 0 Prescale Select

CS02	CS01	CS00	Description
0	0	0	Stop, Timer/Counter0 is stopped.
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	Reserved
1	1	1	Reserved

The Stop condition provides a Timer Enable/Disable function. The prescaled CK modes are scaled directly from the CK oscillator clock.

Timer Counter0 – TCNT0

Bit	7	6	5	4	3	2	1	0	
$32 ($52)	MSB							LSB	TCNT0
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	

The Timer/Counter0 is realized as an up-counter with read and write access. If the Timer/Counter0 is written and a clock source is selected, the Timer/Counter0 continues counting in the timer clock cycle following the write operation.

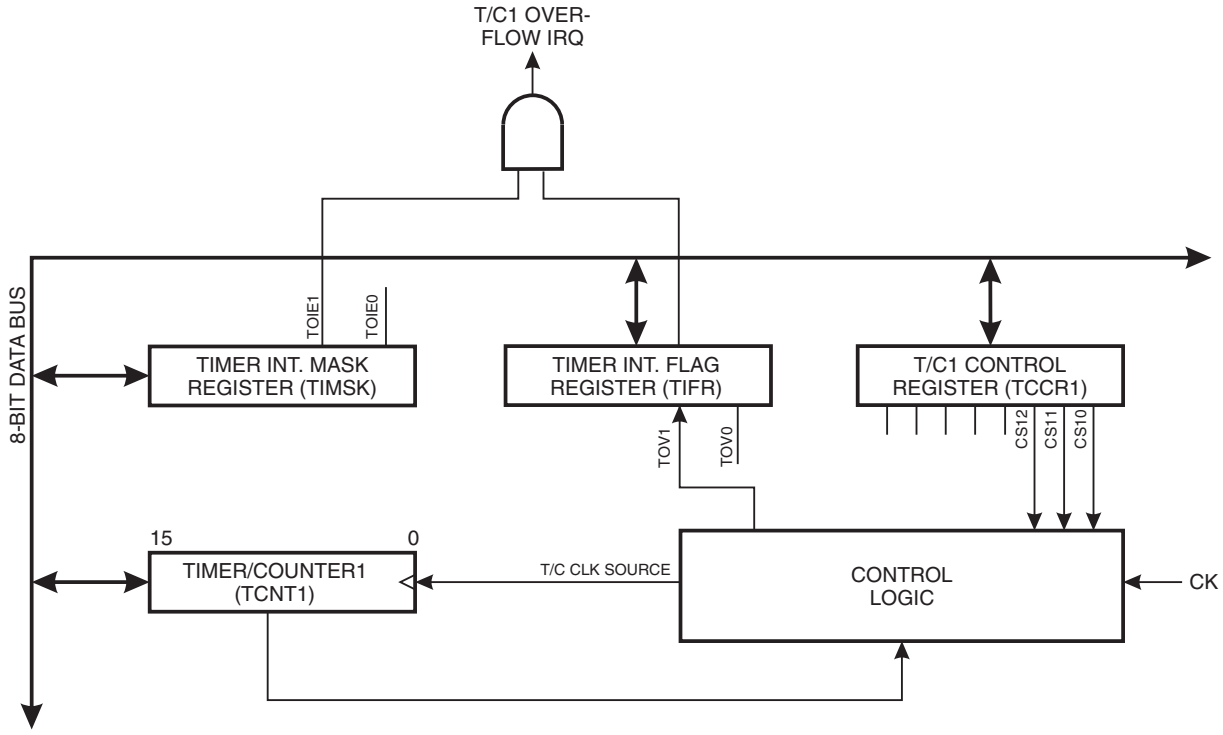
16-bit Timer/Counter1

Figure 28 shows the block diagram for Timer/Counter1.

The 16-bit Timer/Counter1 can select clock source from CK or prescaled CK. In addition, it can be stopped as described in the specification for the Timer/Counter1 Control Register – TCCR1. The overflow status flag is found in the Timer/Counter Interrupt Flag Register – TIFR. Control signals are found in the Timer/Counter1 Control Register – TCCR1. The interrupt enable/disable setting for Timer/Counter1 is found in the Timer/Counter Interrupt Mask Register – TIMSK.

The 16-bit Timer/Counter1 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make the Timer/Counter1 useful for lower speed functions or exact timing functions with infrequent actions.

Figure 28. Timer/Counter1 Block Diagram



Timer/Counter1 Control Register – TCCR1

Bit	7	6	5	4	3	2	1	0	
$2E ($4E)	–	–	–	–	–	CS12	CS11	CS10	TCCR1
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read zero.

- **Bits 2, 1, 0 – CS12, CS11, CS10: Clock Select1, Bits 2, 1 and 0**

The Clock Select1 bits 2, 1 and 0 define the prescaling source of Timer/Counter1.

Table 5. Clock 1 Prescale Select

CS12	CS11	CS10	Description
0	0	0	Stop, Timer/Counter1 is stopped.
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	Reserved
1	1	1	Reserved

The Stop condition provides a Timer Enable/Disable function. The prescaled CK modes are scaled directly from the CK oscillator clock.

Timer/Counter1 – TCNT1H AND TCNT1L

Bit	15	14	13	12	11	10	9	8	
$2D ($4D)	MSB								TCNT1H
$2C ($4C)								LSB	TCNT1L
	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	

This 16-bit register contains the prescaled value of the 16-bit Timer/Counter1. To ensure that both the high and low bytes are read and written simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary register (TEMP). If the main program and interrupt routines perform access using TEMP, interrupts must be disabled during access from the main program.

TCNT1 Timer/Counter1 Write:

When the CPU writes to the high byte TCNT1H, the written data is placed in the TEMP register. Next, when the CPU writes the low byte TCNT1L, this byte of data is combined with the byte data in the TEMP register, and all 16 bits are written to the TCNT1 Timer/Counter1 register simultaneously. Consequently, the high byte TCNT1H must be accessed first for a full 16-bit register write operation.

TCNT1 Timer/Counter1 Read:

When the CPU reads the low byte TCNT1L, the data of the low byte TCNT1L is sent to the CPU and the data of the high byte TCNT1H is placed in the TEMP register. When the CPU reads the data in the high byte TCNT1H, the CPU receives the data in the TEMP register. Consequently, the low byte TCNT1L must be accessed first for a full 16-bit register read operation.

The Timer/Counter1 is realized as an up-counter with read and write access. If Timer/Counter1 is written to and a clock source is selected, the Timer/Counter1 continues counting in the timer clock cycle after it is preset with the written value.

EEPROM Read/Write Access

The EEPROM access registers are accessible in the I/O space.

The write access time is in the range of 2.5 - 35 ms, depending on the V_{CC} voltages. A self-timing function lets the user software detect when the next byte can be written. A special EEPROM Ready interrupt can be set to trigger when the EEPROM is ready to accept new data.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to the description of the EEPROM Control Register for details on this.

When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed. When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed.

EEPROM Address Register – EEARH and EEARL

Bit	15	14	13	12	11	10	9	8	
$1F ($3F)	–	–	–	–	–	–	–	**EEAR9**	**EEARH**
$1E ($3E)	**EEAR7**	**EEAR6**	**EEAR5**	**EEAR4**	**EEAR3**	**EEAR2**	**EEAR1**	**EEAR0**	**EEARL**
	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	x	
	x	x	x	x	x	x	x	x	

The EEPROM Address Registers (EEARH and EEARL) specify the EEPROM address in the 512 bytes EEPROM space. The EEPROM data bytes are addressed linearly between 0 and 511.

EEPROM Data Register – EEDR

Bit	7	6	5	4	3	2	1	0	
$1D ($3D)	**MSB**							**LSB**	**EEDR**
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	

- **Bits 7..0 – EEDR7.0: EEPROM Data**

For the EEPROM write operation, the EEDR register contains the data to be written to the EEPROM in the address given by the EEAR register. For the EEPROM read operation, the EEDR contains the data read out from the EEPROM at the address given by EEAR.

EEPROM Control Register – EECR

Bit	7	6	5	4	3	2	1	0	
$1C ($3C)	–	–	–	–	**EERIE**	**EEMWE**	**EEWE**	**EERE**	**EECR**
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..4 – Res: Reserved Bits**

These bits are reserved bits in the AT90S8535 and will always read as zero.

- **Bit 3 – EERIE: EEPROM Ready Interrupt Enable**

When the I bit in SREG and EERIE are set (one), the EEPROM Ready Interrupt is enabled. When cleared (zero), the interrupt is disabled. The EEPROM Ready interrupt generates a constant interrupt when EEWE is cleared (zero).

- **Bit 2 – EEMWE: EEPROM Master Write Enable**

The EEMWE bit determines whether setting EEWE to one causes the EEPROM to be written. When EEMWE is set (one), setting EEWE will write data to the EEPROM at the selected address. If EEMWE is zero, setting EEWE will have no effect. When EEMWE has been set (one) by software, hardware clears the bit to zero after four clock cycles. See the description of the EEWE bit for a EEPROM write procedure.

- **Bit 1 – EEWE: EEPROM Write Enable**

The EEPROM Write Enable Signal EEWE is the write strobe to the EEPROM. When address and data are correctly set up, the EEWE bit must be set to write the value into the EEPROM. The EEMWE bit must be set when the logical “1” is written to EEWE, otherwise no EEPROM write takes place. The following procedure should be followed when writing the EEPROM (the order of steps 2 and 3 is unessential):

1. Wait until EEWE becomes zero.
2. Write new EEPROM address to EEARL and EEARH (optional).
3. Write new EEPROM data to EEDR (optional).
4. Write a logical “1” to the EEMWE bit in EECR (to be able to write a logical “1” to the EEMWE bit, the EEWE bit must be written to zero in the same cycle).
5. Within four clock cycles after setting EEMWE, write a logical “1” to EEWE.

Caution: An interrupt between step 4 and step 5 will make the write cycle fail, since the EEPROM Master Write Enable will time-out. If an interrupt routine accessing the EEPROM is interrupting another EEPROM access, the EEAR or EEDR register will be modified, causing the interrupted EEPROM access to fail. It is recommended to have the global interrupt flag cleared during steps 2 to 5 to avoid these problems.

When the write access time (typically 2.5 ms at V_{CC} = 5V or 4 ms at V_{CC} = 2.7V) has elapsed, the EEWE bit is cleared (zero) by hardware. The user software should poll this bit and wait for a zero before writing the next byte. When EEWE has been set, the CPU is halted for two cycles before the next instruction is executed.

- **Bit 0 – EERE: EEPROM Read Enable**

The EEPROM Read Enable Signal EERE is the read strobe to the EEPROM. When the correct address is set up in the EEAR register, the EERE bit must be set. When the EERE bit is cleared (zero) by hardware, requested data is found in the EEDR register. The EEPROM read access takes one instruction and there is no need to poll the EERE bit. When EERE has been set, the CPU is halted for four cycles before the next instruction is executed.

The user should poll the EEWE bit before starting the read operation. Writing of any EEPROM I/O register is blocked when a write operation is in progress (except the EERIE bit, which can be written). Hence, if a read access is attempted during a write access, the address cannot be modified and read access will not be performed. The write operation will complete undisturbed.

Prevent EEPROM Corruption

During periods of low V_{CC}, the EEPROM data can be corrupted because the supply voltage is too low for the CPU and the EEPROM to operate properly. These issues are the same as for board-level systems using the EEPROM, and the same design solutions should be applied.

An EEPROM data corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the EEPROM requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly if the supply voltage for executing instructions is too low.

EEPROM data corruption can easily be avoided by following these design recommendations (one is sufficient):

1. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This is best done by an external low V_{CC} Reset Protection circuit, often referred to as a Brown-out Detector (BOD). Please refer to application note AVR 180 for design considerations regarding power-on reset and low-voltage detection.
2. Keep the AVR core in Power-down Sleep Mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the EEPROM registers from unintentional writes.
3. Store constants in Flash memory if the ability to change memory contents from software is not required. Flash memory cannot be updated by the CPU and will not be subject to corruption.

Analog-to-digital Converter

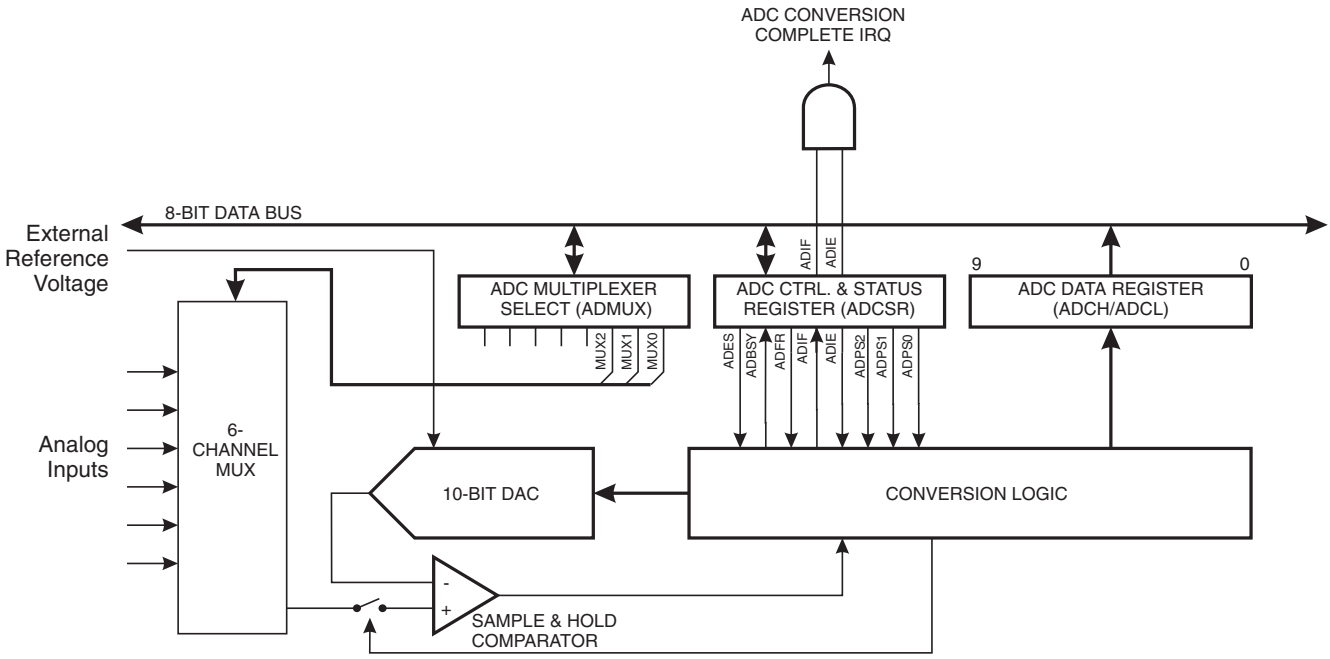
Feature list:

- 10-bit Resolution
- ± 2 LSB Accuracy (AVcc = 3.3 - 6.0V)
- 76 - 175 µs Conversion Time
- Up to 13 kSPS
- 6 Multiplexed Input Channels
- Rail-to-rail Input Range
- Free Run or Single Conversion Mode
- Interrupt on ADC Conversion Complete
- Sleep Mode Noise Canceler

The AT90C8534 features a 10-bit successive approximation ADC. The ADC is connected to a 6-channel Analog Multiplexer, which allows each of the pins ADIN5..0 to be used as an input for the ADC. The ADC contains a Sample and Hold Amplifier that ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown in Figure 29.

The ADC has two separate analog supply voltage pins, AVCC and AGND. AGND must be connected to GND, and the voltage on AVCC must not differ more than ± 0.3V from V_{CC}. See “ADC Noise Canceling Techniques” on page 36 for how to connect these pins.

Figure 29. Analog-to-digital Converter Block Schematic



Operation

The ADC can operate in two modes – Single Conversion and Free Run. In Single Conversion Mode, each conversion will have to be initiated by the user. In Free Run Mode the ADC is constantly sampling and updating the ADC Data Register. The ADFR bit in ADCSR selects between the two available modes.

The ADC is enabled by writing a logical “1” to the ADC Enable bit, ADEN in ADCSR. The first conversion that is started after enabling the ADC will be preceded by a dummy conversion to initialize the ADC. To the user, the only difference will be that this conversion takes 12 more ADC clock pulses than a normal conversion.

A conversion is started by writing a logical “1” to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be set to zero by hardware when the conversion is completed. If a different data channel is selected while a conversion is in progress, the ADC will finish the current conversion before performing the channel change.

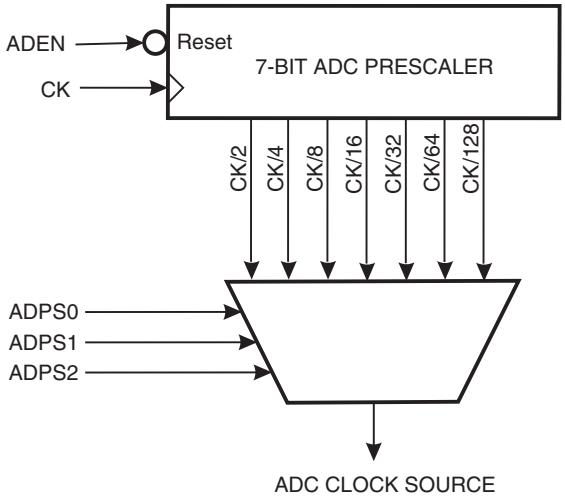
As the ADC generates a 10-bit result, two data registers, ADCH and ADCL, must be read to get the result when the conversion is complete. Special data protection logic is used to ensure that the contents of the data registers belong to the same conversion when they are read. This mechanism works as follows:

When reading data, ADCL must be read first. Once ADCL is read, ADC access to data registers is blocked. This means that if ADCL has been read, and a conversion completes before ADCH is read, none of the registers are updated and the result from the conversion is lost. When ADCH is read, ADC access to the ADCH and ADCL registers is re-enabled.

The ADC has its own interrupt, ADIF, which can be triggered when a conversion completes. When ADC access to the data registers is prohibited between reading of ADCH and ADCL, the interrupt will trigger even if the result gets lost.

Prescaling

Figure 30. ADC Prescaler



The ADC contains a prescaler, which divides the system clock to an acceptable ADC clock frequency. The ADC accepts input clock frequencies in the range of 80 - 170 kHz.

The ADPS0 - ADPS2 bits in ADCSR are used to generate a proper ADC clock input frequency from any XTAL frequency above 160 kHz. The prescaler starts counting from the moment the ADC is switched on by setting the ADEN bit in ADCSR. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low.

When initiating a conversion by setting the ADSC bit in ADCSR, the conversion starts at the following rising edge of the ADC clock cycle. The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of the conversion. The result is ready and written to the ADC Result Register after 13 cycles. In Single Conversion Mode, the ADC needs one more clock cycle before a new conversion can be started (see Figure 32). If ADSC is set high in this period, the ADC will start the new conversion immediately. In Free Run Mode, a new conversion will be started immediately after the result is written to the ADC Result Register. Using Free Run Mode and an ADC clock frequency of 170 kHz gives the lowest conversion time, 76 µs, equivalent to 13 kSPS. For a summary of conversion times, see Table 6.

Figure 31. ADC Timing Diagram, First Conversion (Single Conversion Mode)

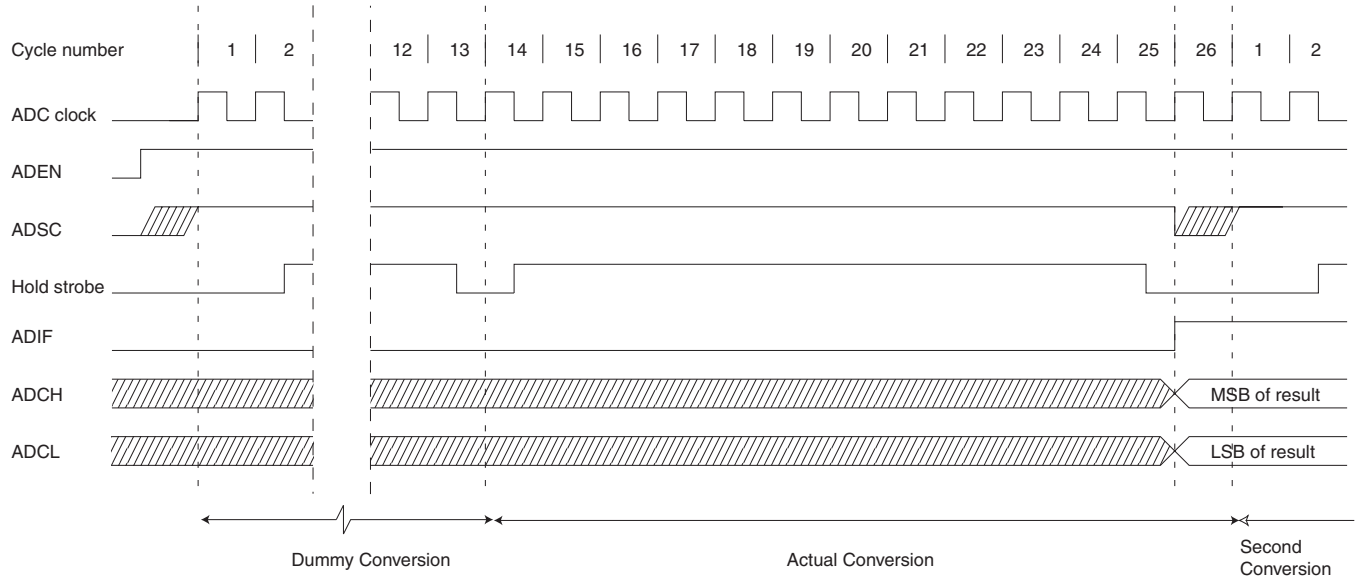


Figure 32. ADC Timing Diagram, Single Conversion

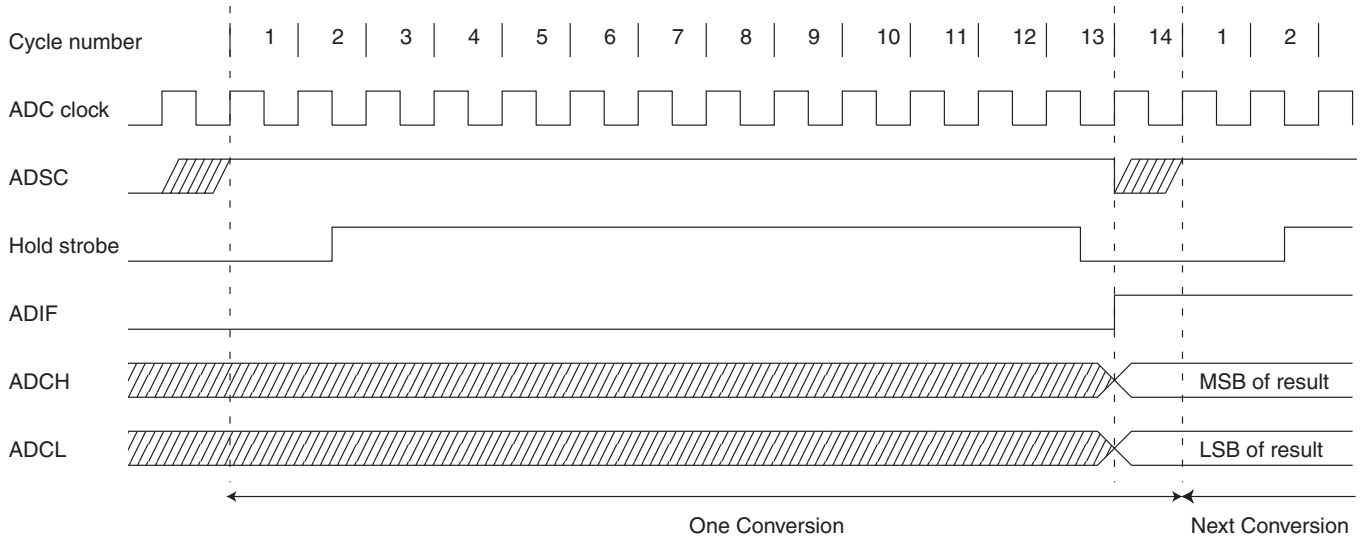


Figure 33. ADC Timing Diagram, Free Run Conversion

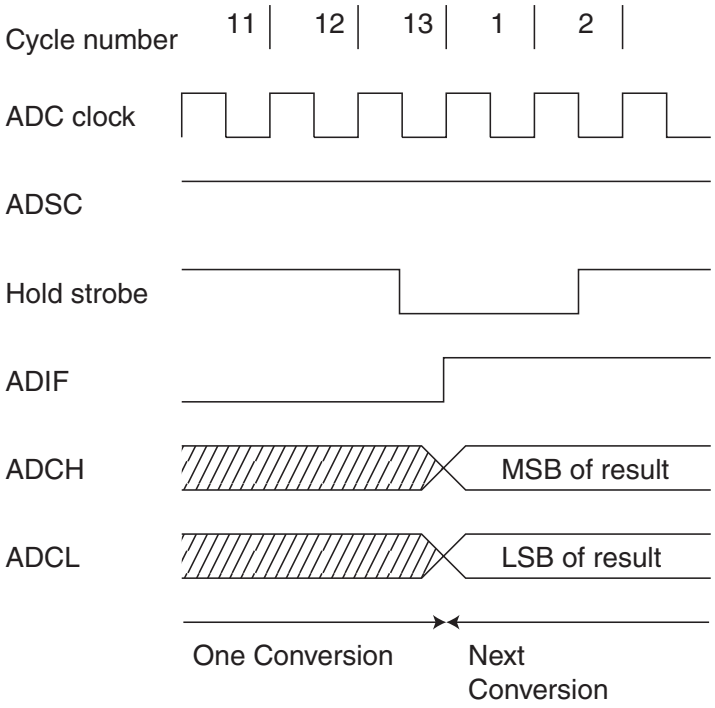


Table 6. ADC Conversion Time

Condition	Sample Cycle Number	Result Ready (Cycle Number)	Total Conversion Time (Cycles)	Total Conversion Time (µs)
1st Conversion, Free Run	14	25	25	147 - 313
1st Conversion, Single	14	25	26	153 - 325
Free Run Conversion	2	13	13	76 - 163
Single Conversion	2	13	14	82 - 175

ADC Noise Canceler Function

The ADC features a noise canceler that enables conversion during idle mode to reduce noise induced from the CPU core. To make use of this feature, the following procedure should be used.

1. Make sure that the ADC is enabled and is not busy converting. Single Conversion Mode must be selected and the ADC conversion complete interrupt must be enabled. Thus:

   ADEN = 1

   ADSC = 0

   ADFR = 0

   ADIE = 1
2. Enter idle mode. The ADC will start a conversion once the CPU has been halted.
3. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the MCU and execute the ADC conversion complete interrupt routine.

ADC Multiplexer Select Register – ADMUX

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	–	–	–	–	–	MUX2	MUX1	MUX0	ADMUX
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the AT90C8534 and always read as zero.

- **Bits 2..0 – MUX2..MUX0: Analog Channel Select Bits 2 - 0**

The value of these three bits selects which analog input 5 - 0 is connected to the ADC. Selections 110 and 111 are reserved and should not be used.

Table 7. ADC Channel Selections

MUX2	MUX1	MUX0	Channel
0	0	0	ADIN0
0	0	1	ADIN1
0	1	0	ADIN2
0	1	1	ADIN3
1	0	0	ADIN4
1	0	1	ADIN5
1	1	0	reserved
1	1	1	reserved

ADC Control and Status Register – ADCSR

Bit	7	6	5	4	3	2	1	0	
$06 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSR
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 – ADEN: ADC Enable**

Writing a logical “1” to this bit enables the ADC. By clearing this bit to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress will terminate this conversion.

- **Bit 6 – ADSC: ADC Start Conversion**

In Single Conversion Mode, a logical “1” must be written to this bit to start each conversion. In Free Run Mode, a logical “1” must be written to this bit to start the first conversion. The first time ADSC has been written after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, a dummy conversion will precede the initiated conversion. This dummy conversion performs initialization of the ADC.

ADSC remains high during the conversion. ADSC goes low after the actual conversion is finished, but before the result is written to the ADC Data Registers. This allows a new conversion to be initiated before the current conversion is complete. The new conversion will then start immediately after the current conversion completes. When a dummy conversion precedes a real conversion, ADSC will stay high until the real conversion is finished.

Writing a 0 to this bit has no effect.

- **Bit 5 – ADFR: ADC Free Run Select**

When this bit is set (one), the ADC operates in Free Run Mode. In this mode, the ADC samples and updates the data registers continuously. Clearing this bit (zero) will terminate Free Run Mode.

- **Bit 4 – ADIF: ADC Interrupt Flag**

This bit is set (one) when an ADC conversion completes and the data registers are updated. The ADC Conversion Complete Interrupt is executed if the ADIE bit and the I-bit in SREG are set (one). ADIF is cleared by hardware when executing

the corresponding interrupt handling vector. Alternatively, ADIF is cleared by writing a logical “1” to the flag. Beware that if doing a read-modify-write on ADCSR, a pending interrupt can be disabled. This also applies if the SBI or CBI instructions are used.

- **Bit 3 – ADIE: ADC Interrupt Enable**

When this bit is set (one) and the I-bit in SREG is set (one), the ADC Conversion Complete Interrupt is activated.

- **Bits 2..0 – ADPS2..ADPS0: ADC Prescaler Select Bits**

These bits determine the division factor between the XTAL frequency and the input clock to the ADC.

Table 8. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

ADC Data Register – ADCL AND ADCH

Bit	15	14	13	12	11	10	9	8	
$05 ($25)	–	–	–	–	–	–	ADC9	ADC8	ADCH
$04 ($24)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial value	8	0	0	0	0	0	0	0	
	8	0	0	0	0	0	0	0	

When an ADC conversion is complete, the result is found in these two registers. In Free Run Mode, it is essential that both registers are read, and that ADCL is read before ADCH.

Scanning Multiple Channels

Since change of analog channel always is delayed until a conversion is finished, the Free Run Mode can be used to scan multiple channels without interrupting the converter. Typically, the ADC Conversion Complete interrupt will be used to perform the channel shift. However, the user should take the following fact into consideration:

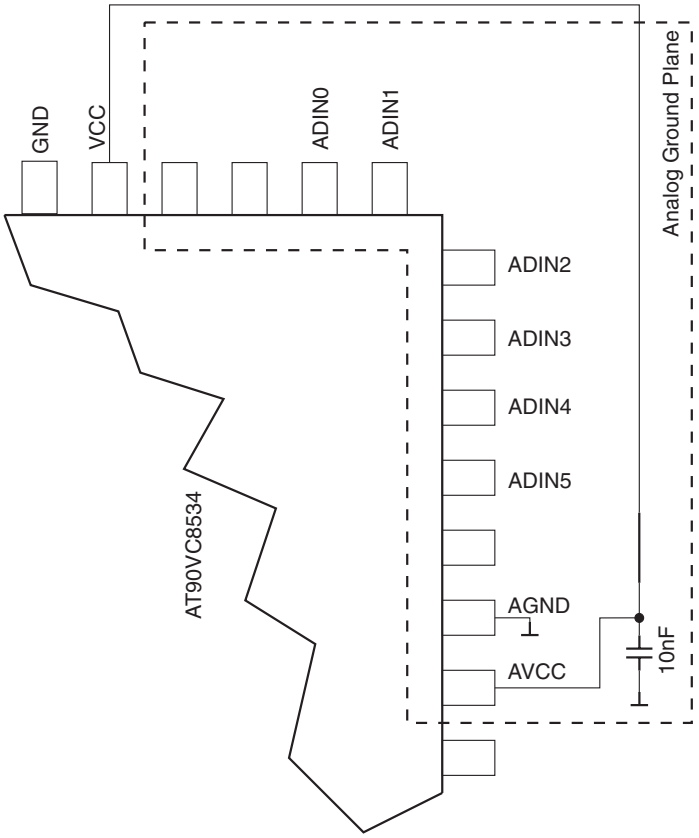
The interrupt triggers once the result is ready to be read. In Free Run Mode, the next conversion will start immediately when the interrupt triggers. If ADMUX is changed after the interrupt triggers, the next conversion has already started and the old setting is used.

ADC Noise Canceling Techniques

Digital circuitry inside and outside the AT90C8534 generates EMI, which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

1. The analog part of the AT90C8534 and all analog components in the application should have a separate analog ground plane on the PCB. This ground plane is connected to the digital ground plane via a single point on the PCB.
2. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.
3. The AVCC pin on the AT90C8534 should be connected to the digital VCC supply voltage as shown in Figure 34.
4. Use the ADC noise canceler function to reduce induced noise from the CPU.

Figure 34. ADC Power Connections



NOTE: PIN PLACEMENT IS AN ILLUSTRATION ONLY

ADC Characteristics

T_A = -40°C to 85°C

Symbol	Parameter	Condition	Min	Typ	Max	Units
	Resolution			10		Bits
	Absolute Accuracy	AV_{CC} = 3.3 - 6.0V			2	LSB
INL	Integral Nonlinearity	AV_{CC} = 3.3 - 6.0V		1		LSB
DNL	Differential Nonlinearity	AV_{CC} = 3.3 - 6.0V		2		LSB
	Zero Error (Offset)	AV_{CC} = 3.3 - 6.0V		0.5		LSB
	Conversion Time		76		175	µs
	Clock Frequency		80		170	kHz
AV_{CC}	Analog Supply Voltage		V_{CC} - 0.3(1)		V_{CC} + 0.3(1)	V
R_{REF}	Reference Input Resistance		6	10	13	KΩ
R_{AIN}	Analog Input Resistance			100		MΩ

Note: 1. AV_{CC} must not go below 3.3V or above 6.0V.

Output Port A

Port A is a 7-bit general output port with tri-state mode.

The port has true read-modify-write functionality. This means that one port pin can be tri-stated without unintentionally tri-stating any other pin with the SBI and CBI instructions. The same applies for changing drive value.

Two I/O memory address locations are allocated for Port A, one each for the Data Register – PORTA, $1B($3B) and Data Direction Register – DDRA, $1A($3A). Both locations are read/write.

The Port A output buffers can sink 20 mA and thus drive LED displays directly.

Port A Data Register – PORTA

Bit	7	6	5	4	3	2	1	0	
$1B ($3B)	–	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Port A Data Direction Register – DDRA

Bit	7	6	5	4	3	2	1	0	
$1A ($3A)	–	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

All seven pins in Port A have equal functionality.

PAn, General Output pin: The DDAn bit in the DDRA register selects tri-state mode of this pin. If DDAn is set (one), PAn is configured to drive out the value in PORTAn. If DDAn is cleared (zero), PAn is configured as a tri-state pin.

Table 9. DDAn Effects on Port A Pins

DDAn	PORTAn	Comment
0	0	Tri-state (high-Z)
0	1	Tri-state (high-Z)
1	0	Push-pull Zero Output
1	1	Push-pull One Output

Note: n: 6, 5, …, 0, pin number.

Memory Programming

Program and Data Memory Lock Bits

The AT90C8534 MCU provides two Lock bits that can be left unprogrammed (“1”) or can be programmed (“0”) to obtain the additional features listed in Table 10.

Table 10. Lock Bit Protection Modes

Memory Lock Bits			
Mode	LB1	LB2	Protection Type
1	1	1	No memory lock features enabled.
2	0	1	Further programming of the Flash and EEPROM is disabled.
3	0	0	Same as mode 2 and verify is also disabled.

Note: The Lock bits can only be erased with the Chip Erase command.

Signature Bytes

All Atmel microcontrollers have a 3-byte signature code that identifies the device. The three bytes reside in a separate address space.

For the AT90C8534 they are:

1. $00: $1E (indicates manufactured by Atmel)
2. $01: $93 (indicates 8 KB Flash memory)
3. $02: $04 (indicates AT90C8534 device when $01 is $93)

Programming the Flash and EEPROM

Atmel’s AT90C8534 offers 8K bytes of Flash program memory and 512 bytes of EEPROM data memory.

The AT90C8534 is shipped with the on-chip Flash program and EEPROM data memory arrays in the erased state (i.e., contents = $FF) and ready to be programmed. This device supports a parallel programming mode, enabled by the $\overline{\text{PEN}}$ pin.

The program and data memory arrays on the AT90C8534 are programmed byte-by-byte.

Parallel Programming

This section describes how to parallel program and verify Flash program memory, EEPROM data memory and memory Lock bits in the AT90C8534.

Signal Names

In this section, some pins of the AT90C8534 are referenced by signal names describing their function during parallel programming. See Figure 35 and Table 11. Pins not described in Table 11 are referenced by pin names.

The XA1/XA0 pins determines the action executed when the XTAL1 pin is given a positive pulse. The coding is shown in Table 12.

When pulsing $\overline{WR}$ or $\overline{OE}$, the command loaded determines the action executed. The command is a byte where the different bits are assigned functions as shown in Table 13.

Figure 35. Parallel Programming

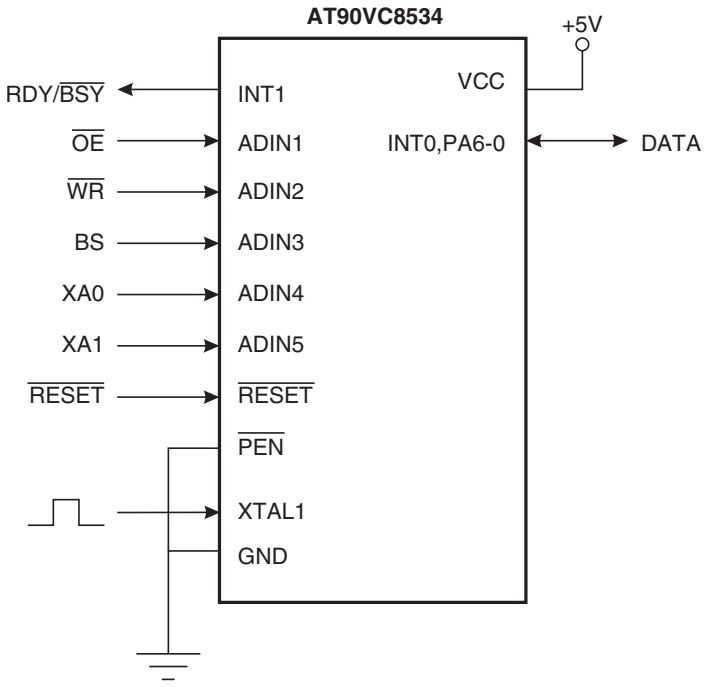


Table 11. Pin Name Mapping

Signal Name in Programming Mode	Pin Name	I/O	Function
RDY/$\overline{BSY}$	INT1	O	"0": Device is busy programming, "1": Device is ready for new command
$\overline{OE}$	ADIN1	I	Output Enable (active low)
$\overline{WR}$	ADIN2	I	Write Pulse (active low)
BS	ADIN3	I	Byte Select ("0" selects low byte, "1" selects high byte)
XA0	ADIN4	I	XTAL1 Action Bit 0
XA1	ADIN5	I	XTAL1 Action Bit 1
DATA	INT0, PA6-0	I/O	Bi-directional Data Bus (output when $\overline{OE}$ is low)

Table 12. XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load Flash/EEPROM/Signature Byte Address (high or low address byte for Flash/EEPROM determined by BS)
0	1	Load Data (high or low data byte for Flash determined by BS)
1	0	Load Command
1	1	No Action, Idle

Table 13. Command Byte Coding

Command Byte	Command Executed
1000 0000	Chip Erase
0010 0000	Write Lock Bits
0001 0000	Write Flash
0001 0001	Write EEPROM
0000 1000	Read Signature Bytes
0000 0100	Read Lock Bits
0000 0010	Read Flash
0000 0011	Read EEPROM

Enter Programming Mode

The following algorithm puts the device in parallel programming mode:

1. Apply 5V between VCC and GND.
2. Set $\overline{\text{PEN}}$, $\overline{\text{RESET}}$ and BS pins to “0” and wait at least 100 ns.
3. Set $\overline{\text{RESET}}$ to “1”. Any activity on BS within 100 ns after $\overline{\text{RESET}}$ is changed to a logical “1” will cause the device to fail entering programming mode.

Chip Erase

The Chip Erase command will erase the Flash and EEPROM memories and the Lock bits. The Lock bits are not reset until the Flash and EEPROM have been completely erased. Chip Erase must be performed before the Flash and EEPROM is reprogrammed.

Load Command “Chip Erase”

1. Set XA1, XA0 to “10”. This enables command loading.
2. Set BS to “0”.
3. Set PB(7:0) to “1000 0000”. This is the command for Chip Erase.
4. Give XTAL1 a positive pulse. This loads the command.
5. Give $\overline{\text{WR}}$ a t_{WLWH_CE} wide negative pulse to execute Chip Erase. See Table 14 for t_{WLWH_CE} value. Chip Erase does not generate any activity on the RDY/$\overline{\text{BSY}}$ pin.

Programming the Flash

A: Load Command “Write Flash”

1. Set XA1, XA0 to “10”. This enables command loading.
2. Set BS to “0”
3. Set DATA to “0001 0000”. This is the command for Write Flash.
4. Give XTAL1 a positive pulse. This loads the command.

B: Load Address High Byte

1. Set XA1, XA0 to “00”. This enables address loading.
2. Set BS to “1”. This selects high byte.
3. Set DATA = Address high byte ($00 - $0F)
4. Give XTAL1 a positive pulse. This loads the address high byte.

C: Load Address Low Byte

1. Set XA1, XA0 to “00”. This enables address loading.
2. Set BS to “0”. This selects low byte.
3. Set DATA = Address low byte ($00 - $FF)
4. Give XTAL1 a positive pulse. This loads the address low byte.

D: Load Data Low Byte

1. Set XA1, XA0 to “01”. This enables data loading.
2. Set DATA = Data low byte ($00 - $FF)
3. Give XTAL1 a positive pulse. This loads the data low byte.

E: Write Data Low Byte

1. Set BS to “0”. This selects low data.
2. Give $\overline{WR}$ a negative pulse. This starts programming of the data byte. RDY/$\overline{BSY}$ goes low.
3. Wait until RDY/$\overline{BSY}$ goes high to program the next byte.

(See Figure 36 for signal waveforms.)

F: Load Data High Byte

1. Set XA1, XA0 to “01”. This enables data loading.
2. Set DATA = Data high byte ($00 - $FF)
3. Give XTAL1 a positive pulse. This loads the data high byte.

G: Write Data High Byte

1. Set BS to “1”. This selects high data.
2. Give $\overline{WR}$ a negative pulse. This starts programming of the data byte. RDY/$\overline{BSY}$ goes low.
3. Wait until RDY/$\overline{BSY}$ goes high to program the next byte.

(See Figure 37 for signal waveforms.)

The loaded command and address are retained in the device during programming. For efficient programming, the following should be considered.

- The command needs only be loaded once when writing or reading multiple memory locations.
- Address high byte needs only be loaded before programming a new 256-word page in the Flash.
- Skip writing the data value $FF, that is, the contents of the entire Flash and EEPROM after a Chip Erase.

These considerations also applies to EEPROM programming, and Flash, EEPROM and signature bytes reading.

Figure 36. Programming the Flash Waveforms

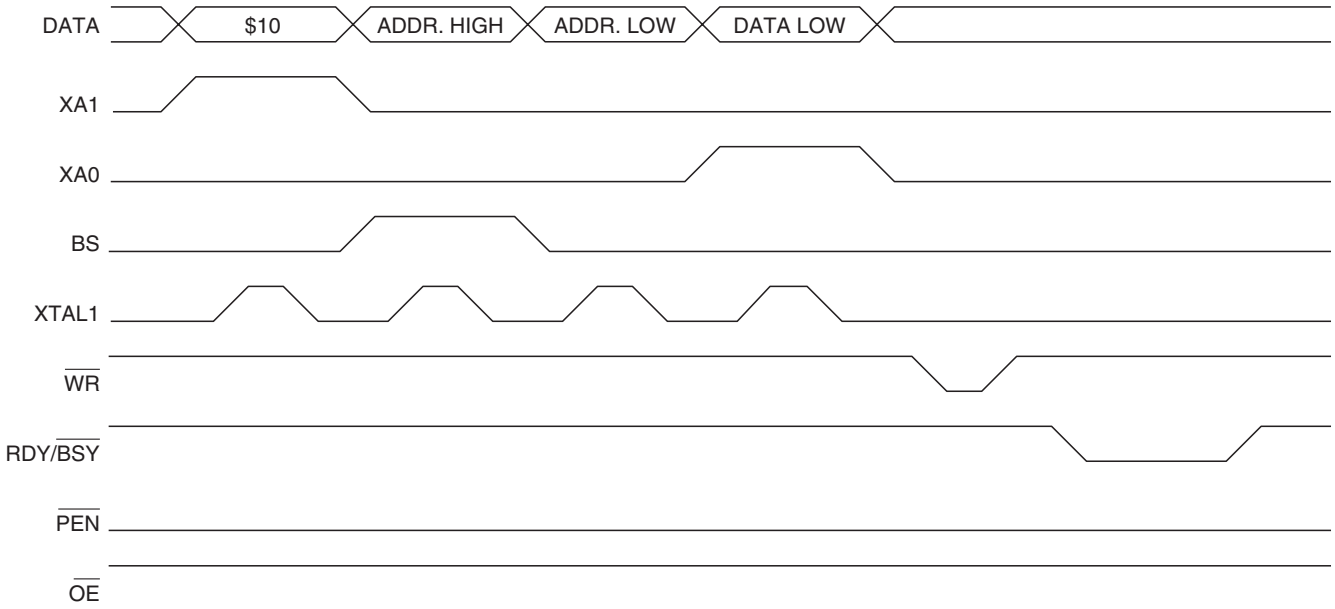
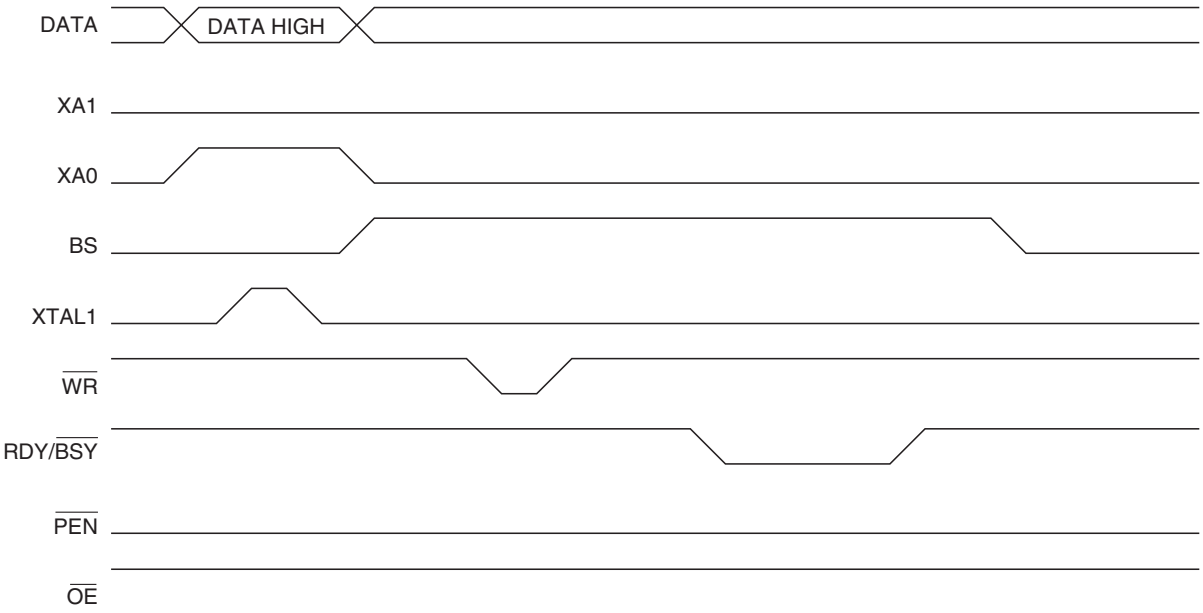


Figure 37. Programming the Flash Waveforms (Continued)



Reading the Flash

The algorithm for reading the Flash memory is as follows (refer to “Programming the Flash” for details on command and address loading):

1. A: Load Command “0000 0010”.
2. B: Load Address High Byte ($00 - $0F).
3. C: Load Address Low Byte ($00 - $FF).
4. Set $\overline{OE}$ to “0”, and BS to “0”. The Flash word low byte can now be read at DATA.
5. Set BS to “1”. The Flash word high byte can now be read from DATA.
6. Set $\overline{OE}$ to “1”.

Programming the EEPROM

The programming algorithm for the EEPROM data memory is as follows (refer to “Programming the Flash” for details on command, address and data loading):

1. A: Load Command “0001 0001”.
2. B: Load Address High Byte ($00 - $01).
3. C: Load Address Low Byte ($00 - $FF).
4. D: Load Data Low Byte ($00 - $FF).
5. E: Write Data Low Byte.

Reading the EEPROM

The algorithm for reading the EEPROM memory is as follows (refer to “Programming the Flash” for details on command and address loading):

1. A: Load Command “0000 0011”.
2. B: Load Address High Byte ($00 - $01).
3. C: Load Address Low Byte ($00 - $FF).
4. Set $\overline{OE}$ to “0”, and BS to “0”. The EEPROM data byte can now be read at DATA.
5. Set $\overline{OE}$ to “1”.

Programming the Lock Bits

The algorithm for programming the Lock bits is as follows (refer to “Programming the Flash” for details on command and data loading):

1. A: Load Command “0010 0000”.
2. D: Load Data Low Byte. Bit n = “0” programs the Lock bit.
   Bit 2 = Lock Bit2
   Bit 1 = Lock Bit1
   Bits 7 - 3, 0 = “1”. These bits are reserved and should be left unprogrammed (“1”).
3. E: Write Data Low Byte.

The Lock bits can only be cleared by executing Chip Erase.

Reading the Lock Bits

The algorithm for reading the Lock bits is as follows (refer to “Programming the Flash” for details on command loading):

1. A: Load Command “0000 0100”.
2. Set $\overline{OE}$ to “0”, and BS to “1”. The status of the Lock bits can now be read at DATA.
   Bit 7: Lock Bit1 (“0” means programmed)
   Bit 6: Lock Bit2 (“0” means programmed)
3. Set $\overline{OE}$ to “1”.

Observe that BS must be set to “1”.

Reading the Signature Bytes

The algorithm for reading the signature bytes is as follows (refer to “Programming the Flash” for details on command and address loading):

1. A: Load Command “0000 1000”.
2. C: Load Address Low Byte ($00 - $02).
   Set $\overline{OE}$ to “0”, and BS to “0”. The selected signature byte can now be read at DATA.
3. Set $\overline{OE}$ to “1”.

Parallel Programming Characteristics

Figure 38. Parallel Programming Timing

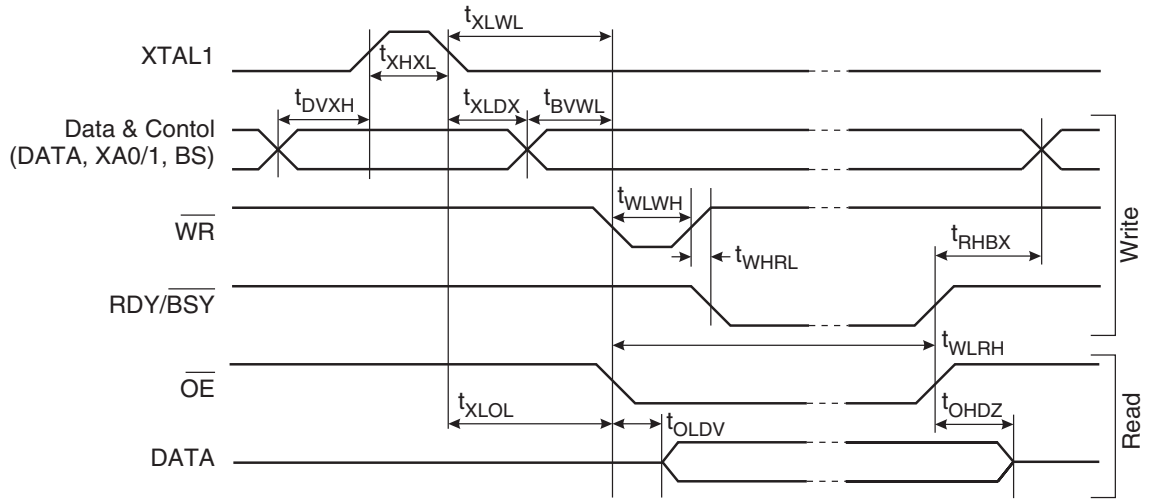


Table 14. Parallel Programming Characteristics

T_A = 25°C ± 10%, V_{CC} = 5.0V ± 10%

Symbol	Parameter	Min	Typ	Max	Units
t_{DVXH}	Data and Control Valid before XTAL1 High	67			ns
t_{XHXL}	XTAL1 Pulse Width High	67			ns
t_{XLDX}	Data and Control Hold after XTAL1 Low	67			ns
t_{XLWL}	XTAL1 Low to $\overline{WR}$ Low	67			ns
t_{BVWL}	BS Valid to $\overline{WR}$ Low	67			ns
t_{RHBX}	BS Hold after RDY/$\overline{BSY}$ High	67			ns
t_{WLWH}	$\overline{WR}$ Pulse Width Low(1)	67			ns
t_{WHRL}	$\overline{WR}$ High to RDY/$\overline{BSY}$ Low(2)		20		ns
t_{WLRH}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ High(2)	0.5	0.7	0.9	ms
t_{XLOL}	XTAL1 Low to $\overline{OE}$ Low	67			ns
t_{OLDV}	$\overline{OE}$ Low to DATA Valid		20		ns
t_{OHDZ}	$\overline{OE}$ High to DATA Tri-stated			20	ns
t_{WLWH_CE}	$\overline{WR}$ Pulse Width Low for Chip Erase	5	10	15	ms

Notes: 1. Use t_{WLWH_CE} for Chip Erase.
2. If t_{WLWH} is held longer than t_{WLRH}, no RDY/$\overline{BSY}$ pulse will be seen.

Electrical Characteristics

Absolute Maximum Ratings*

Operating Temperature	-40°C to +105°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin with respect to Ground	-1.0V to V_{CC} + 0.5V
Maximum Operating Voltage	6.6V
I/O Pin Maximum Current	20.0 mA
Maximum Current VCC and GND	100.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

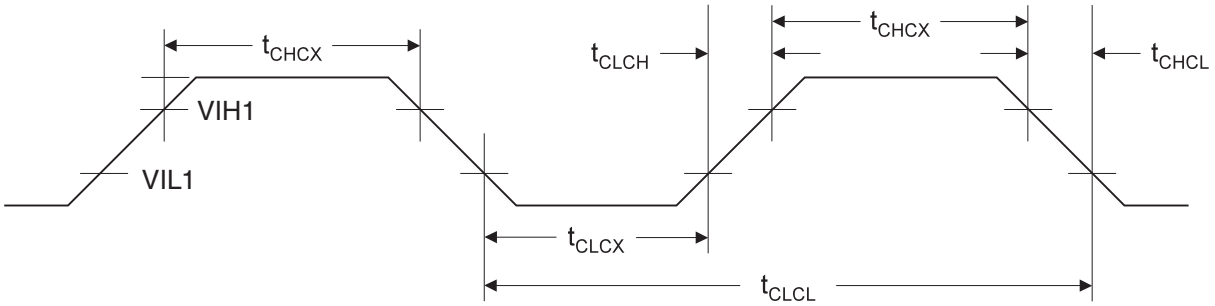
T_A = -40°C to 85°C, V_{CC} = 3.3V to 6.0V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage		-0.5		0.3 V_{CC}[1]	V
V_{IL1}	Input Low Voltage	XTAL	-0.5		0.2 V_{CC}[1]	V
V_{IH}	Input High Voltage	Except XTAL, RESET	0.6 V_{CC}[2]		V_{CC} + 0.5	V
V_{IH1}	Input High Voltage	XTAL	0.8 V_{CC}[2]		V_{CC} + 0.5	V
V_{IH2}	Input High Voltage	RESET	0.9 V_{CC}[2]		V_{CC} + 0.5	V
V_{OL}	Output Low Voltage[3] (Port A)	I_{OL} = 1 mA, V_{CC} = 2.5V			0.1	V
V_{OH}	Output High Voltage[4] (Port A)	I_{OH} = -1 mA, V_{CC} = 2.5V	1.44			V
I_{IL}	Input Leakage Current (I/O pin)	V_{CC} = 6V, pin low	-8.0			µA
I_{IH}	Input Leakage Current (I/O pin)	V_{CC} = 6V, pin high			8.0	µA
RRST	Reset Pull-up		100		500	KΩ
RPEN	PEN Pull-up		30		250	KΩ
I_{CC}	Power Supply Current	Active 1 MHz, V_{CC} = 3.6V, ADC disabled		1.5	2.0	mA
		Active 1 MHz, V_{CC} = 3.6V, ADC enabled		1.9	2.7	mA
		Idle 1 MHz, V_{CC} = 3.6V, ADC disabled		0.25	1.0	mA
		Idle 1 MHz, V_{CC} = 3.6V, ADC enabled		0.7	1.7	mA
		Power-down, V_{CC} = 3.6V		1	10	µA

Notes:
1. "Max" means the highest value where the pin is guaranteed to be read as low (logical "0").
2. "Min" means the lowest value where the pin is guaranteed to be read as high (logical "1").
3. Although each I/O port can sink more than the test conditions (1 mA at V_{CC} = 2.2V) under steady-state conditions (non-transient), the following must be observed:
   The sum of all I_{OL}, for all ports, should not exceed 80 mA.
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
4. Although each I/O port can source more than the test conditions (1 mA at V_{CC} = 2.2V) under steady-state conditions (non--transient), the following must be observed:
   The sum of all I_{OH}, for all ports, should not exceed 80 mA.
   If I_{OH} exceeds the test condition, V_{OH} may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.

External Clock Drive Waveforms

Figure 39. External Clock



External Clock Drive

Symbol	Parameter	V_{CC} = 3.3V to 6.0V		Units
		Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	1.5	MHz
t_{CLCL}	Clock Period	667		ns
t_{CHCX}	High Time	267		ns
t_{CLCX}	Low Time	267		ns
t_{CLCH}	Rise Time		0.5	µs
t_{CHCL}	Fall Time		0.5	µs

Typical Characteristics

The following charts show typical behavior. These data are characterized, but not tested.
Sink and source capabilities of I/O ports are measured on one pin at a time.

Figure 40. I/O Pin Sink Current vs. Output Voltage

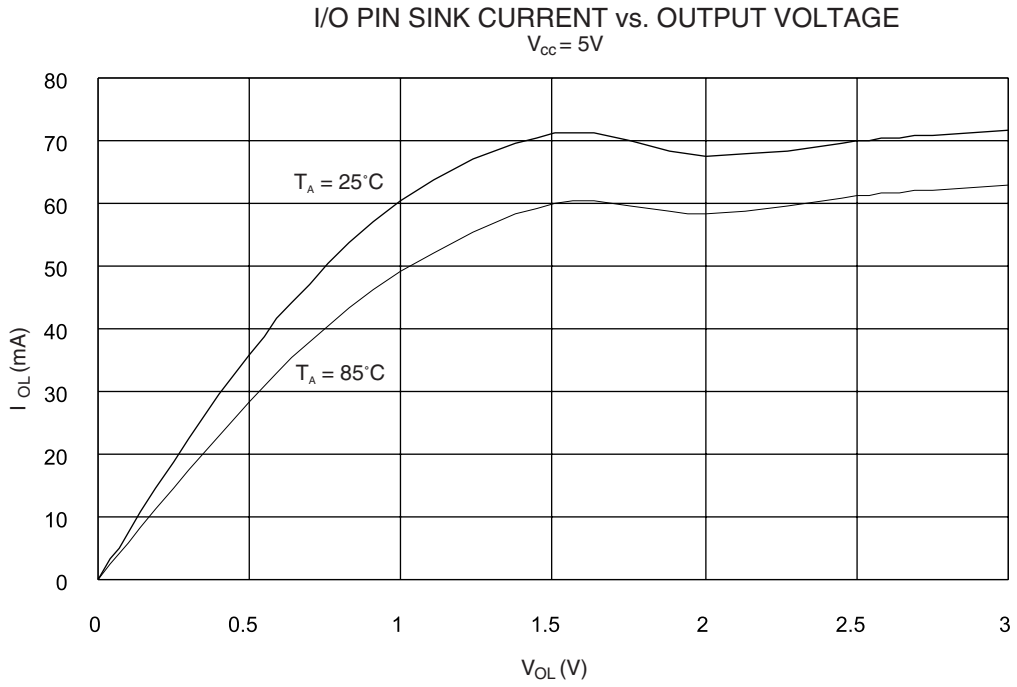


Figure 41. I/O Pin Source Current vs. Output Voltage

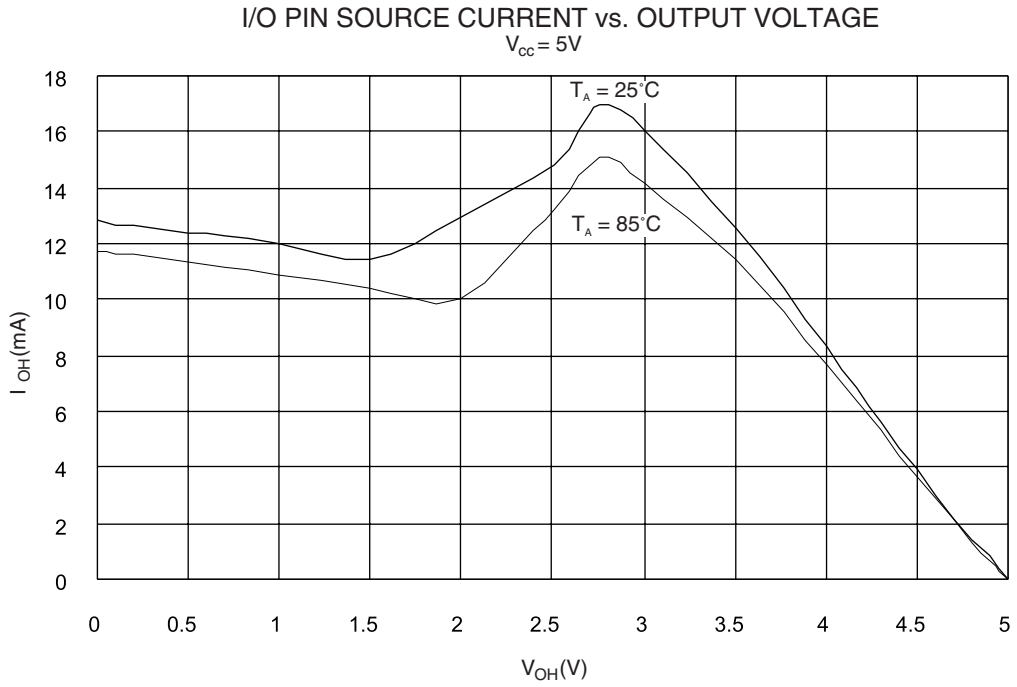


Figure 42. I/O Pin Sink Current vs. Output Voltage

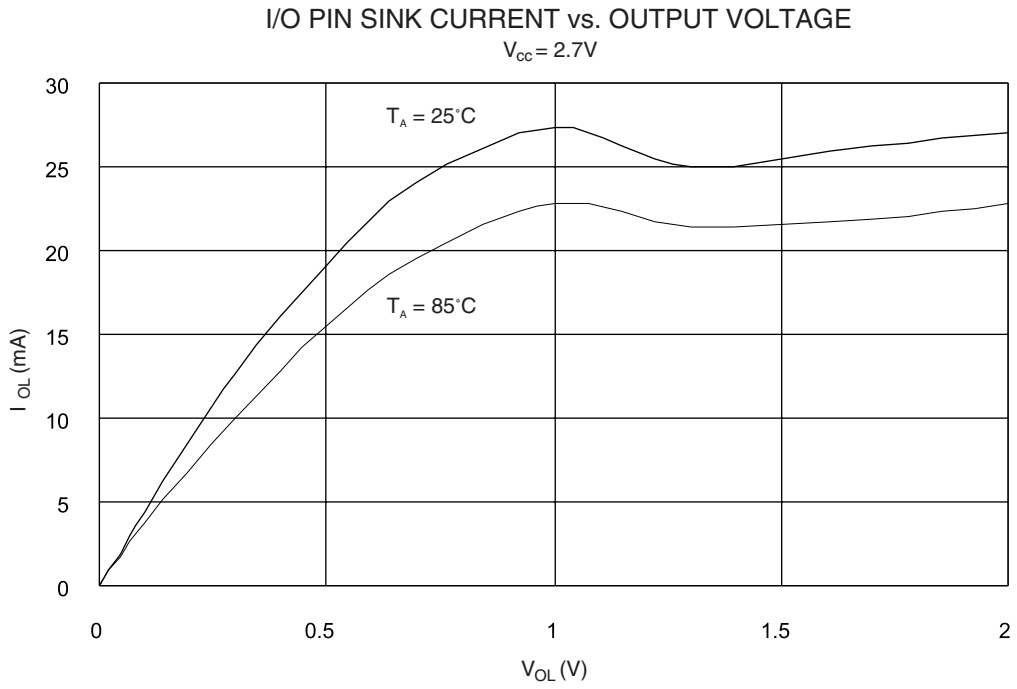
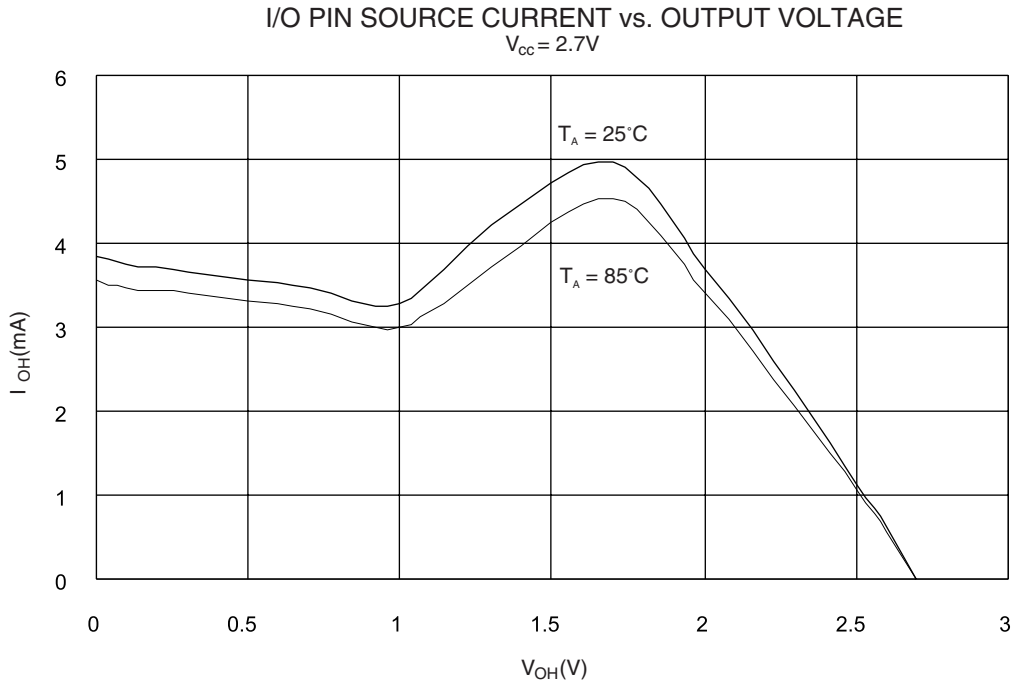


Figure 43. I/O Pin Source Current vs. Output Voltage



AT90C8534 Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	page 17
$3E ($5E)	SPH	-	-	-	-	-	-	-	SP8	page 17
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	page 17
$3C ($5C)	Reserved									
$3B ($5B)	GIMSK	INT1	INT0	-	-	-	-	-	-	page 21
$3A ($5A)	GIFR	INTF1	INTF0							page 21
$39 ($59)	TIMSK	-	-	-	-	-	TOIE1	-	TOIE0	page 22
$38 ($58)	TIFR	-	-	-	-	-	TOV1	-	TOV0	page 22
$37 ($57)	Reserved									
$36 ($56)	Reserved									
$35 ($55)	MCUCR	-	SE	SM	-	-	ISC1	-	ISC0	page 23
$34 ($54)	Reserved									
$33 ($53)	TCCR0	-	-	-	-	-	CS02	CS01	CS00	page 25
$32 ($52)	TCNT0	Timer/Counter0 (8 Bits)								page 26
$31 ($51)	Reserved									
$30 ($50)	Reserved									
$2F ($4F)	Reserved									
$2E ($4E)	TCCR1	-	-	-	-	-	CS12	CS11	CS10	page 26
$2D ($4D)	TCNT1H	Timer/Counter1 - Counter Register High Byte								page 27
$2C ($4C)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								page 27
$2B ($4B)	Reserved									
$2A ($4A)	Reserved									
$29 ($49)	Reserved									
$28 ($48)	Reserved									
$27 ($47)	Reserved									
$26 ($46)	Reserved									
$25 ($45)	Reserved									
$24 ($44)	Reserved									
$23 ($43)	Reserved									
$22 ($42)	Reserved									
$21 ($41)	Reserved									
$20 ($40)	Reserved									
$1F ($3F)	EEARH	-	-	-	-	-	-	-	EEAR8	page 28
$1E ($3E)	EEARL	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	page 28
$1D ($3D)	EEDR	EEPROM Data Register								page 28
$1C ($3C)	EECR	-	-	-	-	EERIE	EEMWE	EEWE	EERE	page 28
$1B ($3B)	PORTA	-	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	page 37
$1A ($3A)	DDRA	-	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	page 37
$19 ($39)	Reserved									
…	Reserved									
$11 ($11)	Reserved									
$10 ($30)	GIPR	-	-	-	-	IPIN1	IPIN0	-	-	page 21
$0F ($2F)	Reserved									
$0E ($2E)	Reserved									
$0D ($2D)	Reserved									
$0C ($2C)	Reserved									
$0B ($2B)	Reserved									
$0A ($2A)	Reserved									
$09 ($29)	Reserved									
$08 ($28)	Reserved									
$07 ($27)	ADMUX	-	-	-	-	-	MUX2	MUX1	MUX0	page 34
$06 ($26)	ADCSR	ADEN	ADSC	ADRF	ADIF	ADIE	ADPS2	ADPS1	ADPS0	page 34
$05 ($25)	ADCH	-	-	-	-	-	-	ADC9	ADC8	page 35
$04 ($24)	ADCL	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	page 35
$03 ($20)	Reserved									
$02 ($22)	Reserved									
$01 ($21)	Reserved									
$00 ($20)	Reserved									

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

Instruction Set Summary

Mnemonic	Operands	Description	Operation	Flags	# Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add Two Registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry Two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl, K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract Two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry Two Registers	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl, K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← $FF - Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← $00 - Rd	Z,C,N,V,H	1
SBR	Rd, K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd, K	Clear Bit(s) in Register	Rd ← Rd • ($FF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd - 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← $FF	None	1
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	3
ICALL		Indirect Call to (Z)	PC ← Z	None	3
RET		Subroutine Return	PC ← STACK	None	4
RETI		Interrupt Return	PC ← STACK	I	4
CPSE	Rd, Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2
CP	Rd, Rr	Compare	Rd - Rr	Z,N,V,C,H	1
CPC	Rd, Rr	Compare with Carry	Rd - Rr - C	Z,N,V,C,H	1
CPI	Rd, K	Compare Register with Immediate	Rd - K	Z,N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b) = 0) PC ← PC + 2 or 3	None	1/2
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b) = 1) PC ← PC + 2 or 3	None	1/2
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b) = 0) PC ← PC + 2 or 3	None	1/2
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b) = 1) PC ← PC + 2 or 3	None	1/2
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC ← PC + k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC ← PC + k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V = 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less than Zero, Signed	if (N ⊕ V = 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half-carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half-carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T-flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T-flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2

Instruction Set Summary (Continued)

Mnemonic	Operands	Description	Operation	Flags	# Clocks
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move between Registers	Rd ← Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, -X	Load Indirect and Pre-dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, -Y	Load Indirect and Pre-dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd, Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-inc.	Rd ← (Z), Z ← Z + 1	None	2
LD	Rd, -Z	Load Indirect and Pre-dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-inc.	(X) ← Rr, X ← X + 1	None	2
ST	-X, Rr	Store Indirect and Pre-dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	-Y, Rr	Store Indirect and Pre-dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q, Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q, Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P, b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P, b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left through Carry	Rd(0) ← C, Rd(n+1) ← Rd(n), C ← Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right through Carry	Rd(7) ← C, Rd(n) ← Rd(n+1), C ← Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n = 0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0) ← Rd(7..4), Rd(7..4) ← Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Two's Complement Overflow	V ← 1	V	1
CLV		Clear Two's Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half-carry Flag in SREG	H ← 1	H	1
CLH		Clear Half-carry Flag in SREG	H ← 0	H	1
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	3
WDR		Watchdog Reset	this command has no effect	None	1

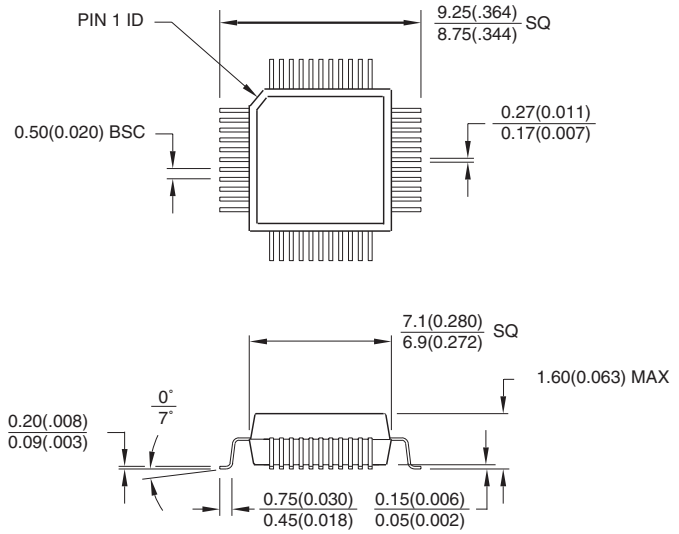
Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
1.5	3.3 - 6.0V	AT90C8534-1AC	48A	Commercial (0°C to 70°C)
1.5	3.3 - 6.0V	AT90C8534-1AI	48A	Industrial (-40°C to 85°C)

Package Type	
48A	48-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)

Packaging Information

48A, 48-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)
Dimensions in Millimeters and (Inches)*



*Controlling dimension: millimeters

Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe
Atmel SarL
Route des Arsenaux 41
Casa Postale 80
CH-1705 Fribourg
Switzerland
TEL (41) 26-426-5555
FAX (41) 26-426-5500

Asia
Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan
Atmel Japan K.K.
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Atmel Colorado Springs
1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset
Zone Industrielle
13106 Rousset Cedex
France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

Atmel Smart Card ICs
Scottish Enterprise Technology Park
East Kilbride, Scotland G75 0QR
TEL (44) 1355-803-000
FAX (44) 1355-242-743

Atmel Grenoble
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex
France
TEL (33) 4-7658-3000
FAX (33) 4-7658-3480

Fax-on-Demand
North America:
1-(800) 292-8635
International:
1-(408) 441-0732

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

BBS
1-(408) 436-4309

1229B–11/00/xM