

Welcome to [E-XFL.COM](https://www.e-xfl.com)

Embedded - FPGAs (Field Programmable Gate Array) with Microcontrollers: Enhancing Flexibility and Performance

Embedded - FPGAs (Field Programmable Gate Arrays) with Microcontrollers represent a cutting-edge category of electronic components that combine the flexibility of FPGA technology with the processing power of integrated microcontrollers. This hybrid approach offers a versatile solution for designing and implementing complex digital systems that require both programmable logic and embedded processing capabilities.

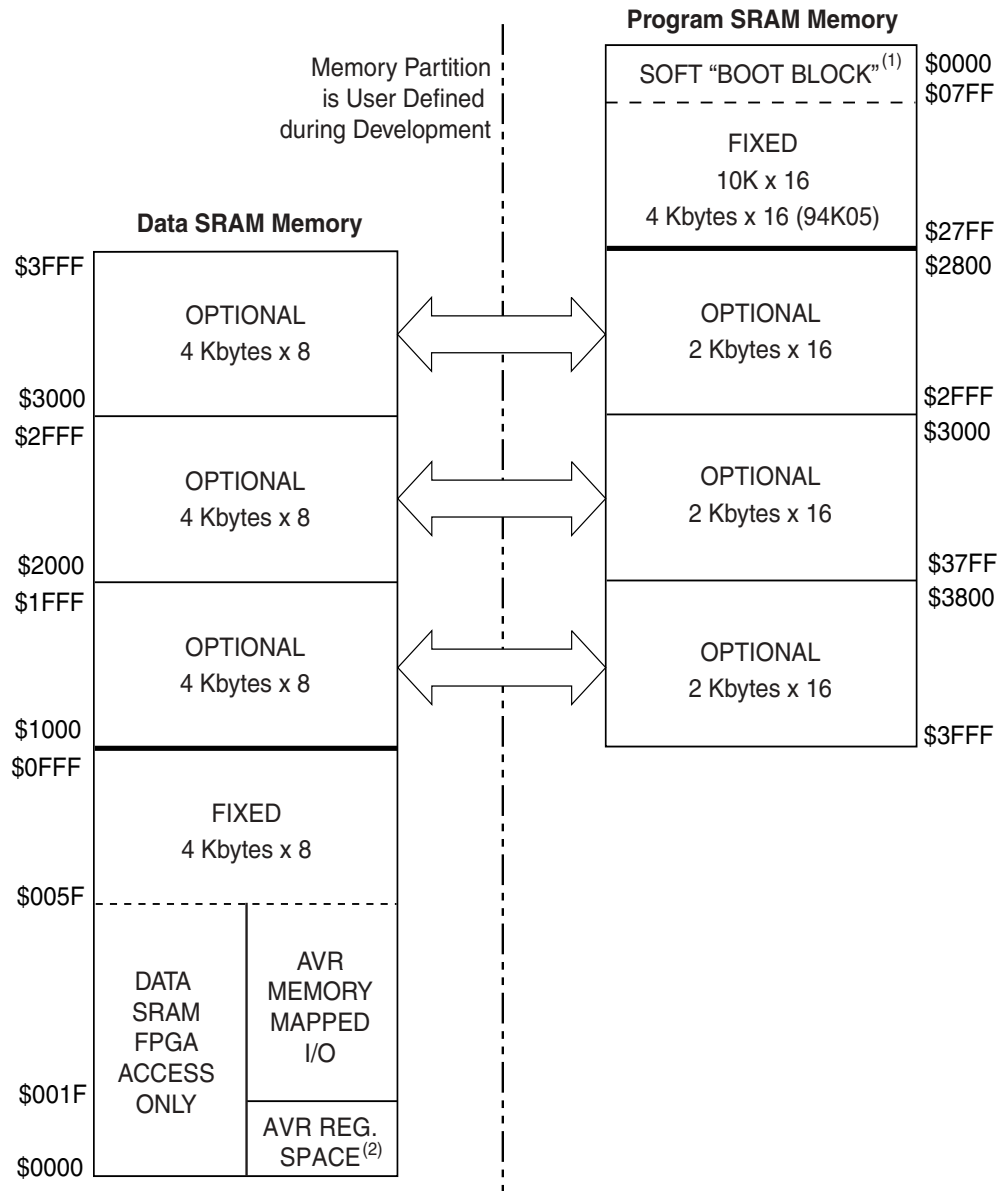
What Are Embedded - FPGAs with Microcontrollers?

At their core, **FPGAs** are semiconductor devices that can

Details

Product Status	Active
Core Type	8-Bit AVR
Speed	25 MHz
Interface	I ² C, UART
Program SRAM Bytes	4K-16K
FPGA SRAM	2kb
EEPROM Size	-
Data SRAM Bytes	4K ~ 16K
FPGA Core Cells	256
FPGA Gates	5K
FPGA Registers	436
Voltage - Supply	3V ~ 3.6V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 85°C
Package / Case	100-TQFP
Supplier Device Package	100-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at94k05al-25aqu

Figure 19. FPSLIC Configurable Allocation SRAM Memory⁽¹⁾⁽²⁾



- Notes:
1. The Soft "BOOT BLOCK" is an area of memory that is first loaded when the part is powered up and configured. The remainder of the memory can be reprogrammed while the device is in operation for switching functions in and out of memory. The Soft "BOOT BLOCK" can only be programmed by a full device configuration on power-up.
 2. The lower portion of the Data memory is not shared between the AVR and FPGA. The AVR uses addresses \$0000 - \$001F for the AVR CPU general working registers. \$001F - \$005F are the addresses used for Memory Mapped I/O and store the information in dedicated registers. Therefore, on the FPGA side \$0000 - \$005F are available for data that is only needed by the FPGA.

A Side

The A side is partitioned into Program memory and Data memory:

- Program memory is 16-bit words.
- Program memory address \$0000 always starts in the highest two SRAMs (n - 1, n) [SRAMn - 1 (low byte) and SRAMn (high byte)] (SRAM labels are for layout, the addressing scheme is transparent to the AVR PC).
- System configuration determines the higher addresses for program memory:
 - SCR bits 41 = 0 : 40 = 0, program memory extended from \$2800 - \$3FFF
 - SCR bits 41 = 0 : 40 = 1, program memory extended from \$2800 - \$37FF
 - SCR bits 41 = 1 : 40 = 0, program memory extended from \$2800 - \$2FFF
 - SCR bits 41 = 1 : 40 = 1, no extra program memory
- Extended program memory is always lost to extended data memory from SRAM2/3 down to SRAM6/7, see Table 5.

Table 5. AVR Program Decode for SRAM 2:7 (16K16)

Address Range	SRAM	Comments
\$3FFF - \$3800	02	CR41:40 = 00
\$3FFF - \$3800	03	
\$37FF - \$3000	04	CR41:40 = 00,01
\$37FF - \$3000	05	
\$2FFF - \$2800	06	CR41:40 = 00,01,10
\$2FFF - \$2800	07	
\$27FF - \$2000	08	AVR Program Read-only
\$27FF - \$2000	09	AVR Program Read-only
\$1FFF - \$1800	10	AVR Program Read-only
\$1FFF - \$1800	11	AVR Program Read-only
\$17FF - \$1000	12	AVR Program Read-only
\$17FF - \$1000	13	AVR Program Read-only
\$0FFF - \$0800	14	AVR Program Read-only
\$0FFF - \$0800	15	AVR Program Read-only
\$07FF - \$0000	16	AVR Program Read-only
\$07FF - \$0000	n = 17	AVR Program Read-only

- Data memory is 8-bit words.
- Data memory address \$0000 always starts in SRAM0 (SRAM labels are for layout, the addressing scheme is transparent to AVR data read/write).
- System configuration determines the higher address for data memory:
 - SCR bits 41 = 0 : 40 = 0, no extra data memory
 - SCR bits 41 = 0 : 40 = 1, data memory extended from \$1000 - \$1FFF
 - SCR bits 41 = 1 : 40 = 0, data memory extended from \$1000 - \$2FFF
 - SCR bits 41 = 1 : 40 = 1, data memory extended from \$1000 - \$3FFF
- Extended data memory is always lost to extended program memory from SRAM7 up to SRAM2 in 2 x SRAM blocks, see Table 6.

Table 11. FPSLIC System Control Register

Bit	Description
SCR6	0 = OTS Disabled 1 = OTS Enabled Setting SCR6 makes the OTS (output tri-state) pin an input which controls the global tri-state control for all user I/O. This junction allows the user at any time to tristate all user I/O and isolate the chip.
SCR7 - SCR12	Reserved
SCR13	0 = CCLK Normal Operation 1 = CCLK Continues After Configuration. Setting bit SCR13 allows the CCLK pin to continue to run after configuration download is completed. This bit is valid for Master mode, mode 0 only. The CCLK is not available internally on the device. If it is required in the design, it must be connected to another device I/O.
SCR14 - SCR15	Reserved
SCR16 - SCR23	0 = GCK 0:7 Always Enabled 1 = GCK 0:7 Disabled During Internal and External Configuration Download. Setting SCR16:SCR23 allows the user to disable the input buffers driving the global clocks. The clock buffers are enabled and disabled synchronously with the rising edge of the respective GCK signal, and stop in a High "1" state. Setting one of these bits disables the appropriate GCK input buffer only and has no effect on the connection from the input buffer to the FPGA array.
SCR24 - SCR25	0 = FCK 0:1 Always Enabled 1 = FCK 0:1 Disabled During Internal and External Configuration Download. Setting SCR24:SCR25 allows the user to disable the input buffers driving the fast clocks. The clock buffers are enabled and disabled synchronously with the rising edge of the respective FCK signal, and stop in a High "1" state. Setting one of these bits disables the appropriate FCK input buffer only and has no effect on the connection from the input buffer to the FPGA array.
SCR26	0 = Disable On-chip Debugger 1 = Enable On-chip Debugger. JTAG Enable, SCR27, must also be set (one) and the configuration memory lockout, SCR4, must be clear (zero) for the user to have access to internal scan chains.
SCR27	0 = Disable TAP at user FPGA I/O Ports 1 = Enable TAP at user FPGA I/O Ports. Device ID scan chain and AVR I/O boundary scan chain are available. The user must set (one) the On-chip Debug Enable, SCR26, and must keep the configuration memory lockout, SCR4, clear (zero) for the user to have access to internal scan chains.
SCR28 - SCR29	Reserved
SCR30	0 = Global Set/Reset Normal 1 = Global Set/Reset Active (Low) During Internal and External Configuration Download. SCR30 allows the Global set/reset to hold the core DFFs in reset during any configuration download. The Global set/reset net is released at the end of configuration download on the rising edge of CON, if set.
SCR31	0 = Disable I/O Tri-state 1 = I/O Tri-state During (Internal and External) Configuration Download. SCR31 forces all user defined I/O pins to go tri-state during configuration download. Tri-state is released at the end of configuration download on the rising edge of CON, if set.



Conditional Branch Summary

Test	Boolean	Mnemonic	Complementary	Boolean	Mnemonic	Comment
Rd > Rr	$Z \cdot (N \oplus V) = 0$	BRLT	Rd ≤ Rr	$Z + (N \oplus V) = 1$	BRGE	Signed
Rd ≥ Rr	$(N \oplus V) = 0$	BRGE	Rd < Rr	$(N \oplus V) = 1$	BRLT	Signed
Rd = Rr	Z = 1	BREQ	Rd ≠ Rr	Z = 0	BRNE	Signed
Rd ≤ Rr	$Z + (N \oplus V) = 1$	BRGE	Rd > Rr	$Z \cdot (N \oplus V) = 0$	BRLT	Signed
Rd < Rr	$(N \oplus V) = 1$	BRLT	Rd ≥ Rr	$(N \oplus V) = 0$	BRGE	Signed
Rd > Rr	C + Z = 0	BRLO	Rd ≤ Rr	C + Z = 1	BRSH	Unsigned
Rd ≥ Rr	C = 0	BRSH/BRCC	Rd < Rr	C = 1	BRLO/BRCS	Unsigned
Rd = Rr	Z = 1	BREQ	Rd ≠ Rr	Z = 0	BRNE	Unsigned
Rd ≤ Rr	C + Z = 1	BRSH	Rd > Rr	C + Z = 0	BRLO	Unsigned
Rd < Rr	C = 1	BRLO/BRCS	Rd ≥ Rr	C = 0	BRSH/BRCC	Unsigned
Carry	C = 1	BRCS	No Carry	C = 0	BRCC	Simple
Negative	N = 1	BRMI	Positive	N = 0	BRPL	Simple
Overflow	V = 1	BRVS	No Overflow	V = 0	BRVC	Simple
Zero	Z = 1	BREQ	Not Zero	Z = 0	BRNE	Simple

Complete Instruction Set Summary

Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clock
Arithmetic and Logic Instructions					
ADD	Rd, Rr	Add without Carry	$Rd \leftarrow Rd + Rr$	Z,C,N,V,S,H	1
ADC	Rd, Rr	Add with Carry	$Rd \leftarrow Rd + Rr + C$	Z,C,N,V,S,H	1
ADIW	Rd, K	Add Immediate to Word	$Rd+1:Rd \leftarrow Rd+1:Rd + K$	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract without Carry	$Rd \leftarrow Rd - Rr$	Z,C,N,V,S,H	1
SUBI	Rd, K	Subtract Immediate	$Rd \leftarrow Rd - K$	Z,C,N,V,S,H	1
SBC	Rd, Rr	Subtract with Carry	$Rd \leftarrow Rd - Rr - C$	Z,C,N,V,S,H	1
SBCI	Rd, K	Subtract Immediate with Carry	$Rd \leftarrow Rd - K - C$	Z,C,N,V,S,H	1
SBIW	Rd, K	Subtract Immediate from Word	$Rd+1:Rd \leftarrow Rd+1:Rd - K$	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND	$Rd \leftarrow Rd \cdot Rr$	Z,N,V,S	1
ANDI	Rd, K	Logical AND with Immediate	$Rd \leftarrow Rd \cdot K$	Z,N,V,S	1
OR	Rd, Rr	Logical OR	$Rd \leftarrow Rd \vee Rr$	Z,N,V,S	1
ORI	Rd, K	Logical OR with Immediate	$Rd \leftarrow Rd \vee K$	Z,N,V,S	1
EOR	Rd, Rr	Exclusive OR	$Rd \leftarrow Rd \oplus Rr$	Z,N,V,S	1
COM	Rd	One's Complement	$Rd \leftarrow \$FF - Rd$	Z,C,N,V,S	1
NEG	Rd	Two's Complement	$Rd \leftarrow \$00 - Rd$	Z,C,N,V,S,H	1
SBR	Rd, K	Set Bit(s) in Register	$Rd \leftarrow Rd \vee K$	Z,N,V,S	1



AT94K Register Summary (Continued)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reference Page
\$16 (\$36)	FISUC	FPGA I/O Select, Interrupt Mask/Flag Register C (Reserved on AT94K05)								54, 56
\$15 (\$35)	FISUB	FPGA I/O Select, Interrupt Mask/Flag Register B								54, 56
\$14 (\$34)	FISUA	FPGA I/O Select, Interrupt Mask/Flag Register A								54, 56
\$13 (\$33)	FISCR	FIADR						XFIS1	XFIS0	53
\$12 (\$32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	124
\$11 (\$31)	DDRD	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0	124
\$10 (\$30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	124
\$0F (\$2F)	Reserved									
\$0E (\$2E)	Reserved									
\$0D (\$2D)	Reserved									
\$0C (\$2C)	UDR0	UART0 I/O Data Register								101
\$0B (\$2B)	UCSR0A	RXC0	TXC0	UDRE0	FE0	OR0		U2X0	MPCM0	101
\$0A (\$2A)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	CHR90	RXB80	TXB80	103
\$09 (\$29)	UBRR0	UART0 Baud-rate Register								105
\$08 (\$28)	OCDR (Reserved)	IDRD								Reserved ⁽¹⁾
\$07 (\$27)	PORTE	PORTE7	PORTE6	PORTE5	PORTE4	PORTE3	PORTE2	PORTE1	PORTE0	126
\$06 (\$26)	DDRE	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0	126
\$05 (\$25)	PINE	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0	126
\$04 (\$24)	Reserved									
\$03 (\$23)	UDR1	UART1 I/O Data Register								101
\$02 (\$22)	UCSR1A	RXC1	TXC1	UDRE1	FE1	OR1		U2X1	MPCM1	101
\$01 (\$21)	UCSR1B	RXCIE1	TXCIE1	UDRIE1	RXEN1	TXEN1	CHR91	RXB81	TXB81	103
\$00 (\$20)	UBRR1	UART1 Baud-rate Register								105

Note: 1. The On-chip Debug Register (OCDR) is detailed on the “FPSLIC On-chip Debug System” distributed within Atmel and select third-party vendors only under Non-Disclosure Agreement (NDA). Contact fpslic@atmel.com for a copy of this document.

The embedded AVR core I/Os and peripherals, and all the virtual FPGA peripherals are placed in the I/O space. The different I/O locations are directly accessed by the IN and OUT instructions transferring data between the 32 x 8 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. When using the I/O specific instructions IN, OUT, the I/O register address \$00 – \$3F are used, see Figure 32. 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.



(FPGAIOWE ← IOWE). FPGA macros utilizing one or more FPGA I/O select lines must use the FPGA I/O read/write enables, FPGAIORE or FPGAIOWE, to qualify each select line. The FIADR bit will be cleared (zero) during AVR reset.

• **Bits 6..2 - Res: Reserved Bits**

These bits are reserved and always read as zero.

• **Bits 1, 0 - XFIS1, 0: Extended FPGA I/O Select Bits 1, 0**

XFIS[1:0] determines which one of the four FPGA I/O select lines will be set (one) within the accessed group. An I/O read or write to one of the four dual-purpose I/O addresses, FISUA..D, will access one of four groups. Table 13 details the FPGA I/O selection scheme.

Table 13. FPGA I/O Select Line Scheme

Read or Write I/O Address	FISCR Register		FPGA I/O Select Lines			
	XFIS1	XFIS0	15..12	11..8	7..4	3..0
FISUA \$14 (\$34)	0	0	0000	0000	0000	0001
	0	1	0000	0000	0000	0010
	1	0	0000	0000	0000	0100
	1	1	0000	0000	0000	1000
FISUB \$15 (\$35)	0	0	0000	0000	0001	0000
	0	1	0000	0000	0010	0000
	1	0	0000	0000	0100	0000
	1	1	0000	0000	1000	0000
FISUC \$16 (\$36) ⁽¹⁾	0	0	0000	0001	0000	0000
	0	1	0000	0010	0000	0000
	1	0	0000	0100	0000	0000
	1	1	0000	1000	0000	0000
FISUD \$17 (\$37) ⁽¹⁾	0	0	0001	0000	0000	0000
	0	1	0010	0000	0000	0000
	1	0	0100	0000	0000	0000
	1	1	1000	0000	0000	0000

Note: 1. Not available on AT94K05.

In summary, 16 select signals are sent to the FPGA for I/O addressing. These signals are decoded from four base I/O Register addresses (FISUA..D) and extended to 16 with two bits from the FPGA I/O Select Control Register, XFIS1 and XFIS0. The FPGA I/O read and write signals, FPGAIORE and FPGAIOWE, are qualified versions of the AVR IORE and IOWE signals. Each will only be active if one of the four base I/O addresses is accessed.

Reset: all select lines become active and an FPGAIOWE strobe is enabled. This is to allow the FPGA design to load zeros (8'h00) from the D-bus into appropriate registers.

FPGA I/O Interrupt Control by AVR

This is an alternate memory space for the FPGA I/O Select addresses. If the FIADR bit in the FISCR register is set to logic 1, the four I/O addresses, FISUA - FISUD, are mapped to physical registers and provide memory space for FPGA interrupt masking and interrupt flag status. If the FIADR bit in the FISCR register is cleared to a logic 0, the I/O register addresses will be decoded into FPGA select lines.

All FPGA interrupt lines into the AVR are negative edge triggered. See page 58 for interrupt priority.

Interrupt Control Registers – FISUA..D

Bit	7	6	5	4	3	2	1	0	
\$14 (\$34)	FIF3	FIF2	FIF1	FIF0	FINT3	FINT2	FINT1	FINT0	FISUA
\$15 (\$35)	FIF7	FIF6	FIF5	FIF4	FINT7	FINT6	FINT5	FINT4	FSUB
\$16 (\$36)	FIF11	FIF10	FIF9	FIF8	FINT11	FINT10	FINT9	FINT8	FISUC
\$17 (\$37)	FIF15	FIF14	FIF13	FIF12	FINT15	FINT14	FINT13	FINT12	FISUD
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..4 - FIF3 - 0: FPGA Interrupt Flags 3 - 0**

The 16 FPGA interrupt flag bits all work the same. Each is set (one) by a valid negative edge transition on its associated interrupt line from the FPGA. Valid transitions are defined as any change in state preceded by at least two cycles of the old state and succeeded by at least two cycles of the new state. Therefore, it is required that interrupt lines transition from 1 to 0 at least two cycles after the line is stable High; the line must then remain stable Low for at least two cycles following the transition. Each bit is cleared by the hardware when executing the corresponding interrupt handling vector. Alternatively, each bit will be cleared by writing a logic 1 to it. When the I-bit in the Status Register, the corresponding FPGA interrupt mask bit and the given FPGA interrupt flag bit are set (one), the associated interrupt is executed.

- **Bits 7..4 - FIF7 - 4: FPGA Interrupt Flags 7 - 4**

See *Bits 7..4 - FIF3 - 0: FPGA Interrupt Flags 3 - 0*.

- **Bits 7..4 - FIF11 - 8: FPGA Interrupt Flags 11 - 8**

See *Bits 7..4 - FIF3 - 0: FPGA Interrupt Flags 3 - 0*. Not available on the AT94K05.

- **Bits 7..4 - FIF15 - 12: FPGA Interrupt Flags 15 - 12**

See *Bits 7..4 - FIF3 - 0: FPGA Interrupt Flags 3 - 0*. Not available on the AT94K05.

- **Bits 3..0 - FINT3 - 0: FPGA Interrupt Masks 3 - 0⁽¹⁾**

The 16 FPGA interrupt mask bits all work the same. When a mask bit is set (one) and the I-bit in the Status Register is set (one), the given FPGA interrupt is enabled. The corresponding interrupt handling vector is executed when the given FPGA interrupt flag bit is set (one) by a negative edge transition on the associated interrupt line from the FPGA.

Note: 1. FPGA interrupts 3 - 0 will cause a wake-up from the AVR Sleep modes. These interrupts are treated as low-level triggered in the Power-down and Power-save modes, see "Sleep Modes" on page 66.

- **Bits 3..0 - FINT7 - 4: FPGA Interrupt Masks 7 - 4**

See *Bits 3..0 - FINT3 - 0: FPGA Interrupt Masks 3 - 0*.

- **Bits 3..0 - FINT11 - 8: FPGA Interrupt Masks 11 - 8**

See *Bits 3..0 - FINT3 - 0: FPGA Interrupt Masks 3 - 0*. Not available on the AT94K05.

- **Bits 3..0 - FINT15 - 12: FPGA Interrupt Masks 15 - 12**

See *Bits 3..0 - FINT3 - 0: FPGA Interrupt Masks 3 - 0*. Not available on the AT94K05.

External Interrupt Mask/Flag Register – EIMF

Bit	7	6	5	4	3	2	1	0	
\$3B (\$5B)	INTF3	INTF2	INTF1	INTF0	INT3	INT2	INT1	INT0	EIMF
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 3..0 - INT3, 2, 1, 0: External Interrupt Request 3, 2, 1, 0 Enable**

When an INT3 - INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the corresponding external pin interrupt is enabled. The external interrupts are always negative edge triggered interrupts, see “Sleep Modes” on page 66.

- **Bits 7..4 - INTF3, 2, 1, 0: External Interrupt 3, 2, 1, 0 Flags**

When a falling edge is detected on the INT3, 2, 1, 0 pins, an interrupt request is triggered. The corresponding interrupt flag, INTF3, 2, 1, 0 becomes set (one). If the I-bit in SREG and the corresponding interrupt enable bit, INT3, 2, 1, 0 in EIMF, are set (one), the MCU will jump to the interrupt vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag is cleared by writing a logic 1 to it.

Timer/Counter Interrupt Mask Register – TIMSK

Bit	7	6	5	4	3	2	1	0	
\$39 (\$39)	TOIE1	OCIE1A	OCIE1B	TOIE2	TICIE1	OCIE2	TOIE0	OCIE0	TIMSK
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 - 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 is executed if an overflow in Timer/Counter1 occurs, i.e., when the TOV1 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 6 - OCIE1A: Timer/Counter1 Output CompareA Match Interrupt Enable**

When the OCIE1A bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 CompareA Match interrupt is enabled. The corresponding interrupt is executed if a CompareA match in Timer/Counter1 occurs, i.e., when the OCF1A bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 5 - OCIE1B: Timer/Counter1 Output CompareB Match Interrupt Enable**

When the OCIE1B bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 CompareB Match interrupt is enabled. The corresponding interrupt is executed if a CompareB match in Timer/Counter1 occurs, i.e., when the OCF1B bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

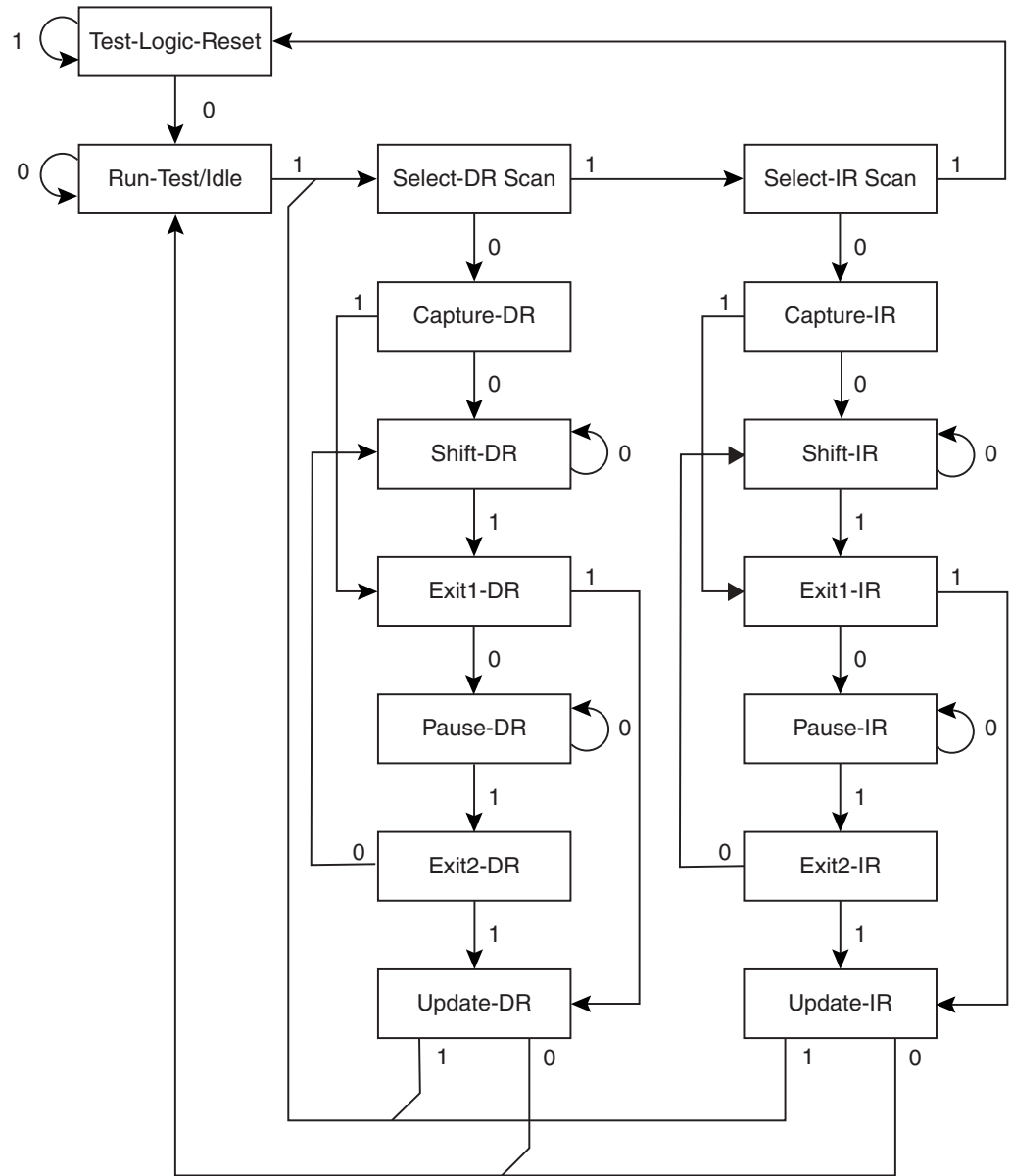
- **Bit 4 - TOIE2: Timer/Counter2 Overflow Interrupt Enable**

When the TOIE2 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter2 overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter2 occurs, i.e., when the TOV2 bit is set in the Timer/Counter interrupt flag register – TIFR.

- **Bit 3 - TICIE1: Timer/Counter1 Input Capture Interrupt Enable**

When the TICIE1 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 input capture event interrupt is enabled. The corresponding interrupt is executed if a capture-triggering event occurs on pin 29, (IC1), i.e., when the ICF1 bit is set in the Timer/Counter interrupt flag register – TIFR.

Figure 40. TAP Controller State Diagram



TAP Controller

The TAP controller is a 16-state finite state machine that controls the operation of the Boundary-Scan circuitry and On-Chip Debug system. The state transitions depicted in Figure 40 depend on the signal present on TMS (shown adjacent to each state transition) at the time of the rising edge at TCK. The initial state after a Power-On Reset is Test-Logic-Reset.

As a definition in this document, the LSB is shifted in and out first for all shift registers.

EXTEST; \$0

Mandatory JTAG instruction for selecting the Boundary-Scan Chain as Data Register for testing circuitry external to the AVR package. For port-pins, Pull-up Disable, Output Control, Output Data, and Input Data are all accessible in the scan chain. For Analog circuits having off-chip connections, the interface between the analog and the digital logic is in the scan chain. The contents of the latched outputs of the Boundary-Scan chain are driven out as soon as the JTAG IR-register is loaded by the EXTEST instruction.

The active states are:

- Capture-DR: Data on the external pins are sampled into the Boundary-Scan Chain.
- Shift-DR: The Internal Scan Chain is shifted by the TCK input.
- Update-DR: Data from the scan chain is applied to output pins.

IDCODE; \$1

Optional JTAG instruction selecting the 32-bit ID register as Data Register. The ID register consists of a version number, a device number and the manufacturer code chosen by JEDEC. This is the default instruction after power-up.

The active states are:

- Capture-DR: Data in the IDCODE register is sampled into the Boundary-Scan Chain.
- Shift-DR: The IDCODE scan chain is shifted by the TCK input.

SAMPLE_PRELOAD; \$2

Mandatory JTAG instruction for pre-loading the output latches and taking a snap-shot of the input/output pins without affecting the system operation. However, the output latches are not connected to the pins. The Boundary-Scan Chain is selected as Data Register.

The active states are:

- Capture-DR: Data on the external pins are sampled into the Boundary-Scan Chain.
- Shift-DR: The Boundary-Scan Chain is shifted by the TCK input.
- Update-DR: Data from the Boundary-Scan chain is applied to the output latches. However, the output latches are not connected to the pins.

AVR_RESET; \$C

The AVR specific public JTAG instruction for forcing the AVR device into the Reset Mode or releasing the JTAG reset source. The TAP controller is not reset by this instruction. The one bit Reset Register is selected as Data Register. Note that the reset will be active as long as there is a logic "1" in the Reset Chain. The output from this chain is not latched.

The active state is:

- Shift-DR: The Reset Register is shifted by the TCK input.

BYPASS; \$F

Mandatory JTAG instruction selecting the Bypass Register for Data Register.

The active states are:

- Capture-DR: Loads a logic "0" into the Bypass Register.
- Shift-DR: The Bypass Register cell between TDI and TDO is shifted.

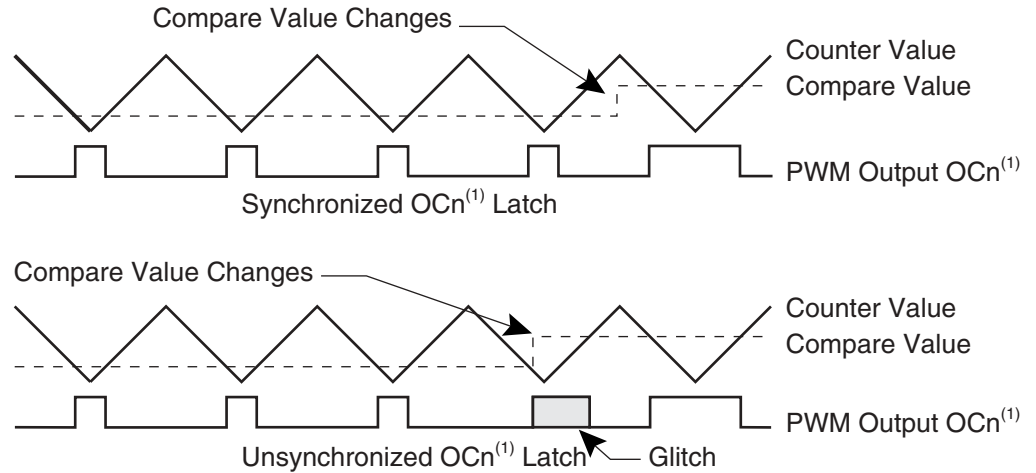
Boundary-scan Chain

The Boundary-Scan chain has the capability of driving and observing the logic levels on the AVR's digital I/O pins.

Scanning the Digital Port Pins

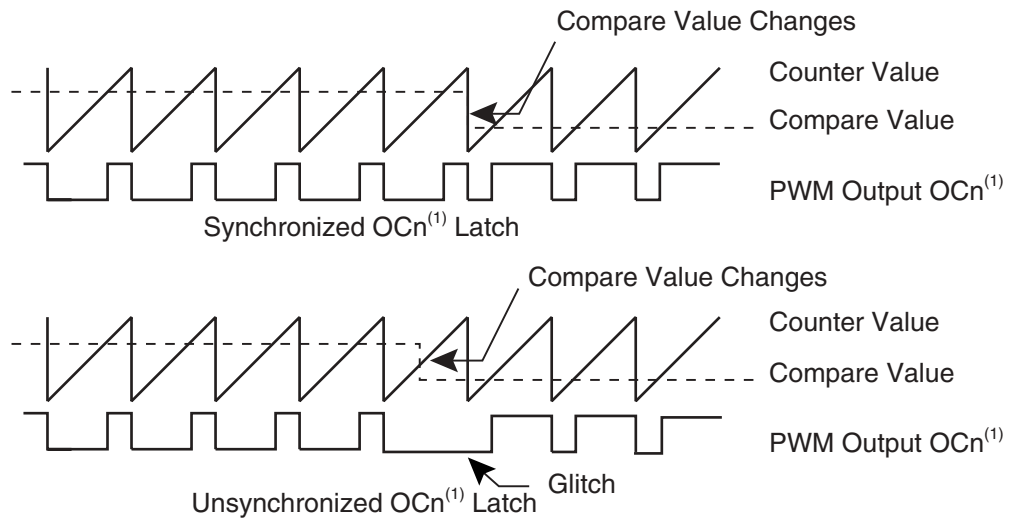
Figure 43 shows the boundary-scan cell for bi-directional port pins with pull-up function. The cell consists of a standard boundary-scan cell for the pull-up function, and a bi-directional pin cell that combines the three signals Output Control (OC), Output Data (OD), and Input Data (ID), into only a two-stage shift register.

Figure 52. Effects of Unsynchronized OCR Latching in Up/Down Mode



Note: 1. n = 0 or 2

Figure 53. Effects of Unsynchronized OCR Latching in Overflow Mode.



Note: 1. n = 0 or 2

During the time between the write and the latch operation, a read from the Output Compare Registers will read the contents of the temporary location. This means that the most recently written value always will read out of OCR0 and OCR2.

When the Output Compare Register contains \$00 or \$FF, and the up/down PWM mode is selected, the output PE1(OC0/PWM0)/PE3(OC2/PWM2) is updated to Low or High on the next compare match according to the settings of COMn1/COMn0. This is shown in Table 26. In overflow PWM mode, the output PE1(OC0/PWM0)/PE3(OC2/PWM2) is held Low or High only when the Output Compare Register contains \$FF.

To convert a negative fractional number, first add 2 to the number and then use the same algorithm as already shown.

16-bit fractional numbers use a format similar to that of 8-bit fractional numbers; the high 8 bits have the same format as the 8-bit format. The low 8 bits are only an increase of accuracy of the 8-bit format; while the 8-bit format has an accuracy of $\pm 2^{-8}$, the 16-bit format has an accuracy of $\pm 2^{-16}$. Then again, the 32-bit fractional numbers are an increase of accuracy to the 16-bit fractional numbers. Note the important difference between integers and fractional numbers when extra byte(s) are used to store the number: while the accuracy of the numbers is increased when fractional numbers are used, the range of numbers that may be represented is extended when integers are used.

As mentioned earlier, using signed fractional numbers in the range $[-1, 1>$ has one main advantage to integers: when multiplying two numbers in the range $[-1, 1>$, the result will be in the range $[-1, 1]$, and an approximation (the highest byte(s)) of the result may be stored in the same number of bytes as the factors, with one exception: when both factors are -1, the product should be 1, but since the number 1 cannot be represented using this number format, the FMULS instruction will instead place the number -1 in R1:R0. The user should therefore assure that at least one of the operands is not -1 when using the FMULS instruction. The 16-bit x 16-bit fractional multiply also has this restriction.

*Example 5 –
Basic Usage
8-bit x 8-bit = 16-bit
Signed Fractional
Multiply*

This example shows an assembly code that reads the port E input value and multiplies this value with a fractional constant (-0.625) before storing the result in register pair R17:R16.

```
in    r16,PINE      ; Read pin values
ldi  r17,$B0       ; Load -0.625 into r17
fmuls r16,r17      ; r1:r0 = r17 * r16
movw r17:r16,r1:r0; Move the result to the r17:r16
                        ; register pair
```

Note that the usage of the FMULS (and FMUL) instructions is very similar to the usage of the MULS and MUL instructions.

Example 6 – Multiply-accumulate Operation

The example below uses data from the ADC. The ADC should be configured so that the format of the ADC result is compatible with the fractional two's complement format. For the ATmega83/163, this means that the ADLAR bit in the ADMUX I/O register is set and a differential channel is used. The ADC result is normalized to one.

```
ldi r23,$62        ; Load highbyte of
                        ; fraction 0.771484375
ldi r22,$C0        ; Load lowbyte of
                        ; fraction 0.771484375
in  r20,ADCL       ; Get lowbyte of ADC conversion
in  r21,ADCH       ; Get highbyte of ADC conversion
callfmac16x16_32   ; Call routine for signed fractional
                        ; multiply accumulate
```

The registers R19:R18:R17:R16 will be incremented with the result of the multiplication of 0.771484375 with the ADC conversion result. In this example, the ADC result is treated as a signed fraction number. We could also treat it as a signed integer and call it "mac16x16_32" instead of "fmac16x16_32". In this case, the 0.771484375 should be replaced with an integer.



```
    mulsu r23, r20          ; (signed)ah * b1
    sbc  r19, r2
    add  r17, r0
    adc  r18, r1
    adc  r19, r2

    mulsu r21, r22          ; (signed)bh * a1
    sbc  r19, r2           ; Sign extend
    add  r17, r0
    adc  r18, r1
    adc  r19, r2

    ret

mac16x16_32_method_B:     ; uses two temporary registers (r4,r5), Speed / Size
                           ; Optimized
                           ; but reduces cycles/words by 1

    clr  r2

    muls r23, r21          ; (signed)ah * (signed)bh
    movw r5:r4,r1:r0

    mul  r22, r20          ; a1 * b1

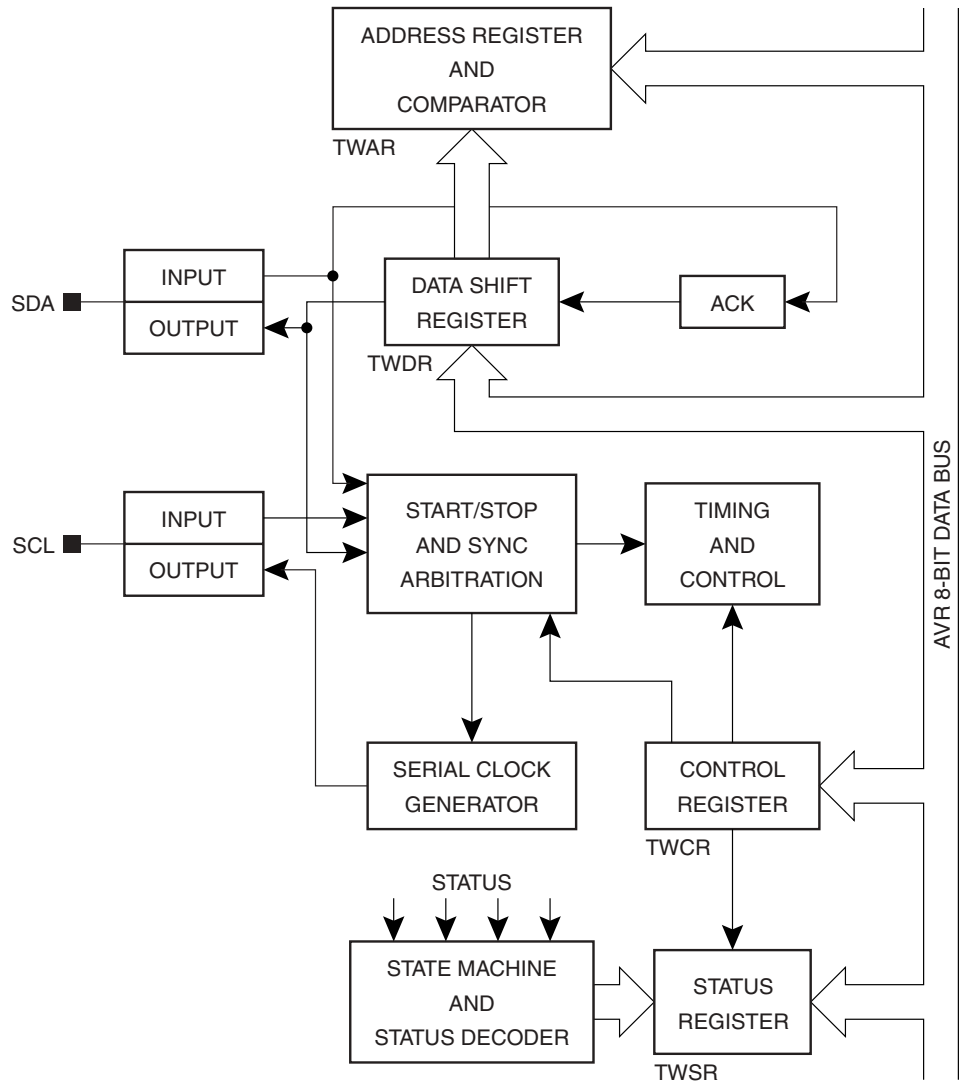
    add  r16, r0
    adc  r17, r1
    adc  r18, r4
    adc  r19, r5

    mulsu r23, r20          ; (signed)ah * b1
    sbc  r19, r2           ; Sign extend
    add  r17, r0
    adc  r18, r1
    adc  r19, r2

    mulsu r21, r22          ; (signed)bh * a1
    sbc  r19, r2           ; Sign extend
    add  r17, r0
    adc  r18, r1
    adc  r19, r2

    ret
```

Figure 70. Block diagram of the 2-wire Serial Bus Interface



The CPU interfaces with the 2-wire Serial Interface via the following five I/O registers: the 2-wire Serial Bit-rate Register (TWBR), the 2-wire Serial Control Register (TWCR), the 2-wire Serial Status Register (TWSR), the 2-wire Serial Data Register (TWDR), and the 2-wire Serial Address Register (TWAR, used in Slave mode).

The 2-wire Serial Bit-rate Register – TWBR

Bit	7	6	5	4	3	2	1	0	
\$1C (\$3C)	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0	TWBR
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 - 2-wire Serial Bit-rate Register**

TWBR selects the division factor for the bit-rate generator. The bit-rate generator is a frequency divider which generates the SCL clock frequency in the Master modes according to the following equation:

$$\text{Bit-rate} = \frac{f_{CK}}{16 + 2(\text{TWBR})}$$

- Bit-rate = SCL frequency
- f_{CK} = CPU Clock frequency
- TWBR = Contents of the 2-wire Serial Bit Rate Register

Both the receiver and the transmitter can stretch the Low period of the SCL line when waiting for user response, thereby reducing the average bit rate.

The 2-wire Serial Control Register – TWCR

Bit	7	6	5	4	3	2	1	0	
\$36 (\$56)	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE	TWCR
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

• **Bit 7 - TWINT: 2-wire Serial Interrupt Flag**

This bit is set by the hardware when the 2-wire Serial Interface has finished its current job and expects application software response. If the I-bit in the SREG and TWIE in the TWCR register are set (one), the MCU will jump to the interrupt vector at address \$0046. While the TWINT flag is set, the bus SCL clock line Low period is stretched. The TWINT flag must be cleared by software by writing a logic 1 to it. Note that this flag is not automatically cleared by the hardware when executing the interrupt routine. Also note that clearing this flag starts the operation of the 2-wire Serial Interface, so all accesses to the 2-wire Serial Address Register – TWAR, 2-wire Serial Status Register – TWSR, and 2-wire Serial Data Register – TWDR must be complete before clearing this flag.

• **Bit 6 - TWEA: 2-wire Serial Enable Acknowledge Flag**

TWEA flag controls the generation of the acknowledge pulse. If the TWEA bit is set, the ACK pulse is generated on the 2-wire Serial Bus if the following conditions are met:

- The device’s own Slave address has been detected
- A general call has been received, while the TWGCE bit in the TWAR is set
- A data byte has been received in Master Receiver or Slave Receiver mode

By setting the TWEA bit Low the device can be virtually disconnected from the 2-wire Serial Bus temporarily. Address recognition can then be resumed by setting the TWEA bit again.

• **Bit 5 - TWSTA: 2-wire Serial Bus START Condition Flag**

The TWSTA flag is set by the CPU when it desires to become a Master on the 2-wire Serial Bus. The 2-wire serial hardware checks if the bus is available, and generates a Start condition on the bus if the bus is free. However, if the bus is not free, the 2-wire Serial Interface waits until a STOP condition is detected, and then generates a new Start condition to claim the bus Master status.

detailed in Table 41. The data must be loaded when TWINT is High only. If not, the access will be discarded, and the Write Collision bit, TWWC, will be set in the TWCR register. This scheme is repeated until a STOP condition is transmitted by writing a logic 1 to the TWSTO bit in the TWCR register.

After a repeated START condition (state \$10) the 2-wire Serial Interface may switch to the Master Receiver mode by loading TWDR with SLA+R.

Master Receiver Mode

In the Master Receiver mode, a number of data bytes are received from a Slave Transmitter, see Figure 72. The transfer is initialized as in the Master Transmitter mode. When the START condition has been transmitted, the TWINT flag is set by the hardware. The software must then load TWDR with the 7-bit Slave address and the data direction bit (SLA+R). The 2-wire Serial Interrupt flag must then be cleared by software before the 2-wire Serial Transfer can continue.

When the Slave address and the direction bit have been transmitted and an acknowledgment bit has been received, TWINT is set again and a number of status codes in TWSR are possible. Status codes \$40, \$48, or \$38 apply to Master mode, and status codes \$68, \$78, or \$B0 apply to Slave mode. The appropriate action to be taken for each of these status codes is detailed in Table 42. Received data can be read from the TWDR register when the TWINT flag is set High by the hardware. This scheme is repeated until a STOP condition is transmitted by writing a logic 1 to the TWSTO bit in the TWCR register.

After a repeated START condition (state \$10), the 2-wire Serial Interface may switch to the Master Transmitter mode by loading TWDR with SLA+W.

Slave Receiver Mode

In the Slave Receiver mode, a number of data bytes are received from a Master Transmitter, see Figure 73. To initiate the Slave Receiver mode, TWAR and TWCR must be initialized as follows:

Table 39. TWAR: Slave Receiver Mode Initialization

TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
value	Device's own Slave address							

The upper 7 bits are the address to which the 2-wire Serial Interface will respond when addressed by a Master. If the LSB is set, the 2-wire Serial Interface will respond to the general call address (\$00), otherwise it will ignore the general call address.

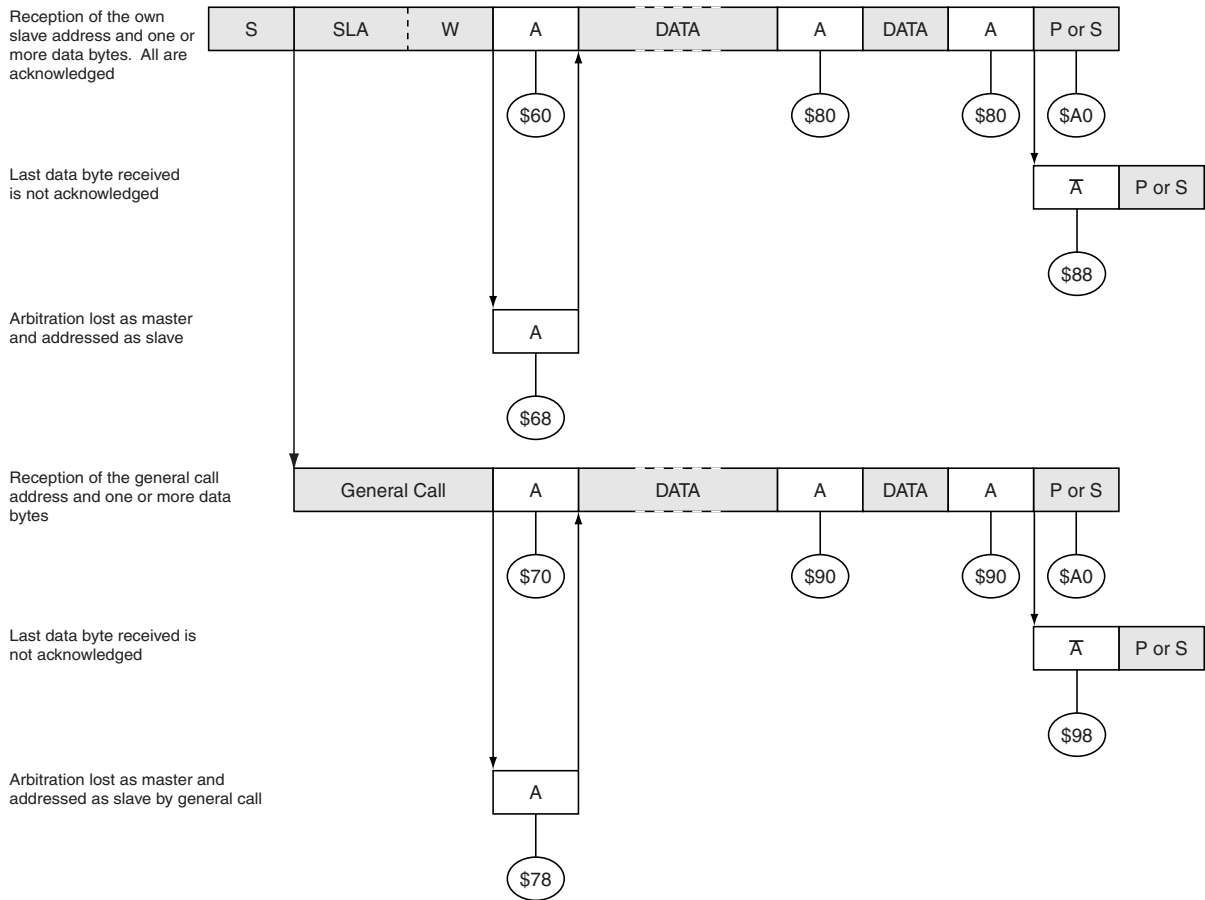
Table 40. TWCR: Slave Receiver Mode Initialization



TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
value	0	1	0	0	0	1	0	X



TWEN must be set to enable the 2-wire Serial Interface. The TWEA bit must be set to enable the acknowledgment of the device's own Slave address or the general call address. TWSTA and TWSTO must be cleared.


When TWAR and TWCR have been initialized, the 2-wire Serial Interface waits until it is addressed by its own Slave address (or the general call address if enabled) followed by the data direction bit which must be "0" (write) for the 2-wire Serial Interface to operate in the Slave Receiver mode. After its own Slave address and the write bit have been received, the 2-wire Serial Interrupt flag is set and a valid status code can be read from TWSR. The status code is used to determine the appropriate software action. The appropriate action to be taken for each status code is detailed in Table 43. The Slave Receiver mode may also be entered if arbitration is lost while the 2-wire Serial Interface is in the Master mode (see states \$68 and \$78).

Figure 73. Formats and States in the Slave Receiver Mode



 From master to slave
 From slave to master

 DATA
 A
 Any number of data bytes and their associated acknowledge bits

 n
 This number (contained in TWSR) corresponds to a defined state of the 2-wire serial bus

AC & DC Timing Characteristics

Absolute Maximum Ratings^{*(1)}

Operating Temperature.....	-55°C to +125°C
Storage Temperature.....	-65°C to +150°C
Voltage ⁽²⁾ on Any Pin with Respect to Ground.....	-0.5V to +5.0V
Supply Voltage (V _{CC}).....	-0.5V to +5.0V
Maximum Soldering Temp. (10 sec. @ 1/16 in.).....	250°C
ESD (R _{ZAP} = 1.5K, C _{ZAP} = 100 pF).....	2000V

***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 any other conditions beyond those listed under operating conditions is not implied. Exposure to Absolute Maximum Rating conditions for extended periods of time may affect device reliability.

- Notes: 1. For AL parts only
2. Minimum voltage of -0.5V DC which may undershoot to -2.0V for pulses of less than 20 ns.

DC and AC Operating Range – 3.3V Operation

		AT94K Commercial	AT94K Industrial
Operating Temperature (Case)		0°C - 70°C	-40°C - 85°C
V _{CC} Power Supply		3.3V ± 0.3V	3.3V ± 0.3V
Input Voltage Level (CMOS)	High (V _{IHC})	70% - 100% V _{CC}	70% - 100% V _{CC}
	Low (V _{ILC})	0 - 30% V _{CC}	0 - 30% V _{CC}

Table 52. FPSLIC Interface Timing Information⁽¹⁾

Symbol	Parameter	3.3V Commercial \pm 10%			3.3V Industrial \pm 10%			Units
		Minimum	Typical	Maximum	Minimum	Typical	Maximum	
t_{IXG4}	Clock Delay From XTAL2 Pad to GCK_5 Access to FPGA Core	3.6	4.8	7.6	3.4	4.8	7.9	ns
t_{IXG5}	Clock Delay From XTAL2 Pad to GCK_6 Access to FPGA Core	3.9	5.2	8.1	3.6	5.2	8.8	ns
t_{IXC}	Clock Delay From XTAL2 Pad to AVR Core Clock	2.8	3.7	6.3	2.5	3.7	6.9	ns
t_{IXI}	Clock Delay From XTAL2 Pad to AVR I/O Clock	3.5	4.7	7.5	3.2	4.7	7.8	ns
t_{CFIR}	AVR Core Clock to FPGA I/O Read Enable	5.3	6.6	7.9	4.4	6.6	9.2	ns
t_{CFIW}	AVR Core Clock to FPGA I/O Write Enable	5.2	6.6	7.9	4.4	6.6	9.2	ns
t_{CFIS}	AVR Core Clock to FPGA I/O Select Active	6.3	7.8	9.4	5.3	7.8	11.0	ns
t_{FIRQ}	FPGA Interrupt Net Propagation Delay to AVR Core	0.2	0.2	0.3	0.1	0.2	0.3	ns
t_{IFS}	FPGA SRAM Clock to On-chip SRAM	6.1	7.7	7.7	4.9	7.7	7.7	ns
t_{FRWS}	FPGA SRAM Write Strobe to On-chip SRAM	4.4	5.5	5.5	3.7	5.5	5.5	ns
t_{FAS}	FPGA SRAM Address Valid to On-chip SRAM Address Valid	5.4	6.7	6.7	4.3	6.7	6.7	ns
t_{FDWS}	FPGA Write Data Valid to On-chip SRAM Data Valid	1.3	1.7	2.0	1.3	1.7	2.0	ns
t_{FDRS}	On-chip SRAM Data Valid to FPGA Read Data Valid	0.2	0.2	0.2	0.2	0.2	0.2	ns

Note: 1. Insertion delays are specified from XTAL2. These delays are more meaningful because the XTAL1-to-XTAL2 delay is sensitive to system loading on XTAL2. If it is necessary to drive external devices with the system clock, devices should use XTAL2 output pin. Remember that XTAL2 is inverted in comparison to XTAL1.



Table 56. AT94K Pin List (Continued)

AT94K05 96 FPGA I/O	AT94K10 192 FPGA I/O	AT94K40 384 FPGA I/O	Packages			
			PC84	TQ100	PQ144	PQ208
VCC ⁽¹⁾	VCC ⁽¹⁾	VCC ⁽¹⁾	54	51	73	106
$\overline{\text{RESET}}$	$\overline{\text{RESET}}$	$\overline{\text{RESET}}$	55	52	74	108
PE0	PE0	PE0	56	53	75	109
PE1	PE1	PE1	57	54	76	110
PD0	PD0	PD0			77	111
PD1	PD1	PD1			78	112
		GND				
		VCC ⁽¹⁾				
		GND				
PE2	PE2	PE2	58	55	79	113
PD2	PD2	PD2		56	80	114
		GND				
No Connect	No Connect	No Connect			81	119
PD3	PD3	PD3			82	120
PD4	PD4	PD4			83	121
	VCC ⁽¹⁾	VCC ⁽¹⁾				
PE3	PE3	PE3	59	57	84	122
$\overline{\text{CS}}_0, \text{Cs}0n$	$\overline{\text{CS}}_0, \text{Cs}0n$	$\overline{\text{CS}}_0, \text{Cs}0n$	60	58	85	123
		GND				
		GND				
		VCC ⁽¹⁾				
SDA	SDA	SDA				124
SCL	SCL	SCL				125
		GND				
PD5	PD5	PD5		59	86	126
PD6	PD6	PD6		60	87	127
PE4	PE4	PE4	61	61	88	128
PE5	PE5	PE5	62	62	89	129
VDD ⁽²⁾	VDD ⁽²⁾	VDD ⁽²⁾	63	63	90	130
GND	GND	GND	64	64	91	131
PE6	PE6	PE6	65	65	92	132
PE7 ($\overline{\text{CHECK}}$)	PE7 ($\overline{\text{CHECK}}$)	PE7 ($\overline{\text{CHECK}}$)	66	66	93	133
PD7	PD7	PD7		67	94	134

Notes: 1. VCC is I/O high voltage. Please refer to the “Designing in Split Power Supply Support for AT94KAL and AT94SAL Devices” application note.
2. VDD is core high voltage. Please refer to the “Designing in Split Power Supply Support for AT94KAL and AT94SAL Devices” application note.
3. Unbonded pins are No Connects.