Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	AVR
Core Size	8-Bit
Speed	4MHz
Connectivity	SPI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	7
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 9V
Data Converters	A/D 5x12b
Oscillator Type	Internal
Operating Temperature	-20°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	36-WFLGA
Supplier Device Package	36-LGA (6.5x3.5)
Purchase URL	https://www.e-xfl.com/product-detail/atmel/atmega16hva-4cku

The FCAL[4:0] bits are used to tune the frequency within the selected range. A setting of 0x00 gives the lowest frequency in that range, and a setting of 0x1F gives the highest frequency in the range. Incrementing FCAL[4:0] by 1 will give a frequency increment of less than 1.5 % in the frequency range 7.3-8.1 MHz. With an accurate time reference, an oscillator accuracy of ±1% can be achieved after calibration. The frequency will drift with temperature, so run-time calibration will be required to maintain the accuracy. Refer to ”OSI – Oscillator Sampling Interface” on page 28 for details.

The default FOSCCAL value found in the signature row, is selected such that it is in the the lower half of a segment (see Figure 30-1 on page 174 for typical characteristics of the FAST RC oscillator). It is therefore sufficient to use the default segment and the one below to calibrate the frequency over the whole temperature range. To avoid a large frequency change when shifting between the two segments, a FOSC SEGMENT value is stored in the signature row. This is the first FOSCCAL value giving a lower frequency than the lowest value in the default segment, and should be used when calibrating the Fast RC oscillator.

9.13.2 MCUCR – MCU Control Register

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	-–	–	CKOE	PUD	–	–	–	–	MCUCR
Read/Write	R	R	R/W	R/W	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 5 – CKOE: Clock Output**

When this bit is written to one, the CPU clock divided by 2 is output on the PB0 pin.

9.13.3 CLKPR – Clock Prescale Register

Bit	7	6	5	4	3	2	1	0	
(0x61)	CLKPCE	–	–	–	–	–	CLKPS1	CLKPS0	CLKPR
Read/Write	R/W	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	1	1	

- **Bit 3 - PRSPI: Power Reduction Serial Peripheral Interface**

Writing logic one to this bit shuts down the Serial Peripheral Interface by stopping the clock to the module. When waking up the SPI again, the SPI should be reinitialized to ensure proper operation.

- **Bit 2 - PRTIM1: Power Reduction Timer/Counter1**

Writing a logic one to this bit shuts down the Timer/Counter1 module. When the Timer/Counter1 is enabled, operation will continue like before the shutdown.

- **Bit 1 - PRTIM0: Power Reduction Timer/Counter0**

Writing a logic one to this bit shuts down the Timer/Counter0 module. When the Timer/Counter0 is enabled, operation will continue like before the shutdown.

- **Bit 0 - PRVADC: Power Reduction V-ADC**

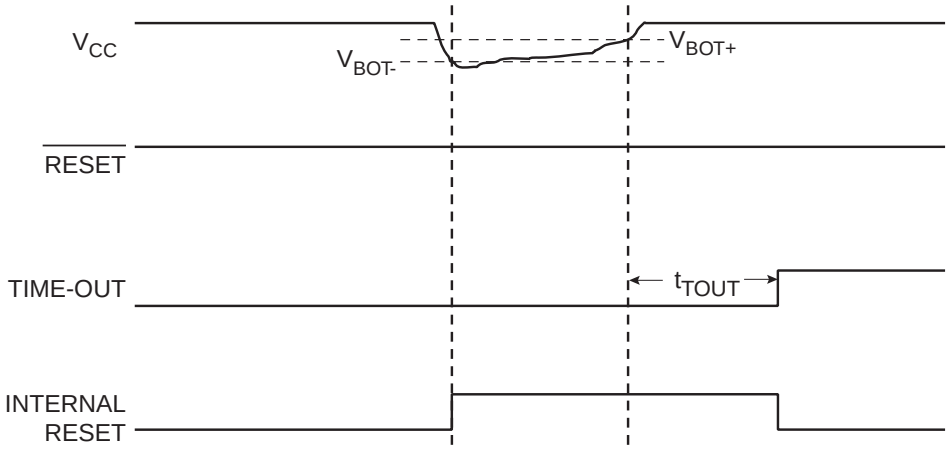
Writing a logic one to this bit shuts down the V-ADC. Before writing the PRVADC bit, make sure that the VADEN bit is cleared to minimize the power consumption.

Note: V-ADC control registers can be updated even if the PRVADC bit is set.

The BOD is automatically enabled in all modes of operation, except in Power-off mode.

When the BOD is enabled, and V_{REG} decreases to a value below the trigger level (V_{BOT-} in Figure 11-5), the Brown-out Reset is immediately activated. When V_{REG} increases above the trigger level (V_{BOT+} in Figure 11-5), the delay counter starts the MCU after the Time-out period t_{TOUT} has expired.

Figure 11-5. Brown-out Reset During Operation



11.2.5 Black-out Detection

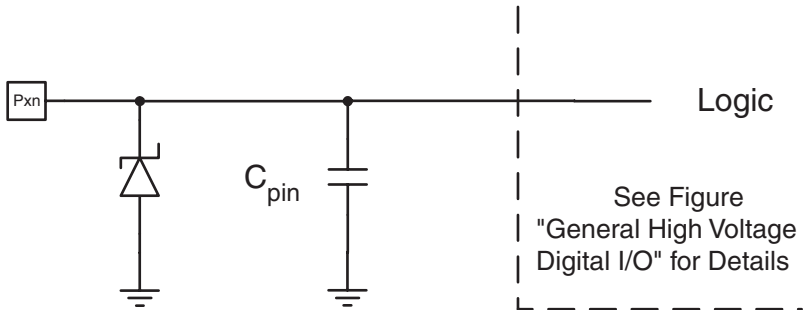
As an extra security feature, the chip will automatically enter Power-off if V_{REG} drops below V_{BLOT}. V_{BLOT} will always be well below the BOD level, V_{BOT}.

14. High Voltage I/O Ports

14.1 Overview

All high voltage AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the state of one port pin can be changed without unintentionally changing the state of any other pin with the SBI and CBI instructions. All high voltage I/O pins have protection Zener diodes to Ground as indicated in Figure 14-1. See ”Electrical Characteristics” on page 165 for a complete list of parameters.

Figure 14-1. High Voltage I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case “x” represents the numbering letter for the port, and a lower case “n” represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. For example, PORTC3 for bit number three in Port C, here documented generally as PORTxn. The physical I/O Registers and bit locations are listed in ”Register Description” on page 62.

One I/O Memory address location is allocated for each high voltage port, the Data Register – PORTx. The Data Register is read/write.

Using the I/O port as General Digital Output is described in ”High Voltage Ports as General Digital I/O” on page 59.

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

15.2.2 Toggling the Pin

Writing a logic one to PINxn toggles the value of PORTxn, independent on the value of DDRxn. Note that the SBI instruction can be used to toggle one single bit in a port.

15.2.3 Switching Between Input and Output

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled {DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedant environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the MCUCR Register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b11) as an intermediate step.

Table 15-1 summarizes the control signals for the pin value.

Table 15-1. Port Pin Configurations

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

15.2.4 Reading the Pin Value

Independent of the setting of Data Direction bit DDxn, the port pin can be read through the PINxn Register bit. As shown in Figure 15-2, the PINxn Register bit and the preceding latch constitute a synchronizer. This is needed to avoid metastability if the physical pin changes value near the edge of the internal clock, but it also introduces a delay. Figure 15-3 shows a timing diagram of the synchronization when reading an externally applied pin value. The maximum and minimum propagation delays are denoted $t_{pd,max}$ and $t_{pd,min}$ respectively.

16. Timer/Counter0 and Timer/Counter1 Prescalers

16.1 Overview

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

16.1.1 Internal Clock Source

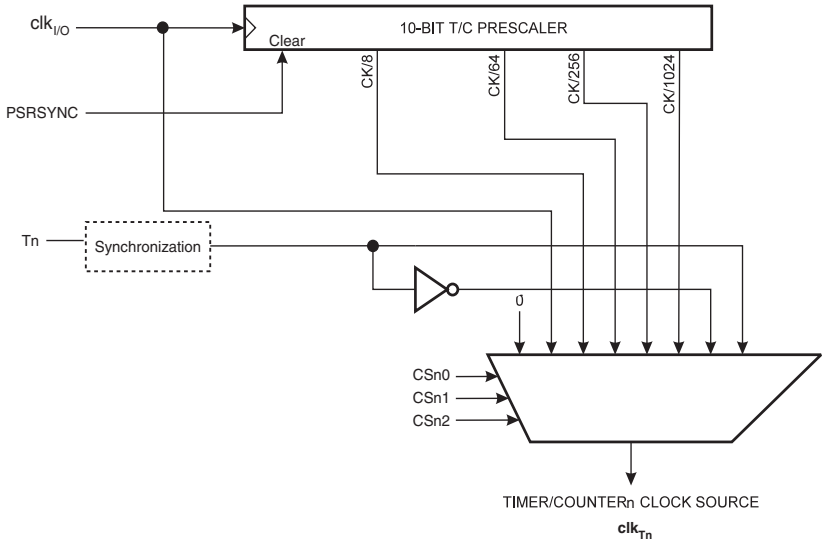
The Timer/Counter can be clocked directly by the system clock (by setting the CSn2:0 = 1). This provides the fastest operation, with a maximum Timer/Counter clock frequency equal to system clock frequency ($f_{CLK\_I/O}$). Alternatively, one of four taps from the prescaler can be used as a clock source.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

16.1.2 Prescaler Reset

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

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

Figure 16-1. Prescaler for Timer/Counter



16.3 Register Description

16.3.1 TCCRnB – Timer/Counter n Control Register B

Bit	7	6	5	4	3	2	1	0	
	-	-	-	-	-	CSn2	CSn1	CSn0	TCCRnB
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 2, 1, 0 – CSn2, CSn1, CSn0: Clock Select n, Bit 2, 1, and 0**

The Clock Select n bits 2, 1, and 0 define the prescaling source of Timer n.

Table 16-1. Clock Select Bit Description

CSn2	CSn1	CSn0	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on Tn pin. Clock on falling edge.
1	1	1	External clock source on Tn pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter n, transitions on the Tn pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

16.3.2 General Timer/Counter Control Register – GTCCR

Bit	7	6	5	4	3	2	1	0	
	TSM	–	–	–	–	–	–	PSRSYNC	GTCCR
Read/Write	R/W	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – TSM: Timer/Counter Synchronization Mode**

Writing the TSM bit to one activates the Timer/Counter Synchronization mode. In this mode, the value that is written to the PSRSYNC bit is kept, hence keeping the corresponding prescaler reset signals asserted. This ensures that the corresponding Timer/Counters are halted and can be configured to the same value without the risk of one of them advancing during configuration. When the TSM bit is written to zero the PSRSYNC bit is cleared by hardware, and the Timer/Counters start counting simultaneously.

- **Bit 0 – PSRSYNC: Prescaler Reset**

When this bit is one, Timer/Counter1 and Timer/Counter0 prescaler will be reset. This bit is normally cleared immediately by hardware, except if the TSM bit is set. Note that Timer/Counter1 and Timer/Counter0 share the same prescaler and a reset of this prescaler will affect both timers.

The following code examples show how to access the 16-bit timer registers assuming that no interrupts updates the temporary register. The same principle can be used directly for accessing the OCRnB/A registers.

Assembly Code Example

```
...
; Set TCNTn to 0x01FF
ldi r17,0x01
ldi r16,0xFF
out TCNTnH,r17
out TCNTnL,r16
; Read TCNTn into r17:r16
in  r16,TCNTnL
in  r17,TCNTnH
...
```

C Code Example

```
unsigned int i;
...
/* Set TCNTn to 0x01FF */
TCNTn = 0x1FF;
/* Read TCNTn into i */
i = TCNTn;
...
```

Note: 1. See “About Code Examples” on page 7.

The assembly code example returns the TCNTnH/L value in the r17:r16 register pair.

It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register, and the interrupt code updates the temporary register by accessing the same or any other of the 16-bit timer registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the temporary register, the main code must disable the interrupts during the 16-bit access.

Table 18-1. SPI Pin Overrides[1]

Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

Note: 1. See "Alternate Functions of Port B" on page 71 for a detailed description of how to define the direction of the user defined SPI pins.

The following code examples show how to initialize the SPI as a Master and how to perform a simple transmission. DDR_SPI in the examples must be replaced by the actual Data Direction Register controlling the SPI pins. DD_MOSI, DD_MISO and DD_SCK must be replaced by the actual data direction bits for these pins. E.g. if MOSI is placed on pin PB5, replace DD_MOSI with DDB5 and DDR_SPI with DDRB.

Assembly Code Example[1]

```
SPI_MasterInit:
  ; Set MOSI and SCK output, all others input
  ldi  r17,(1<<DD_MOSI)|(1<<DD_SCK)
  out  DDR_SPI,r17
  ; Enable SPI, Master, set clock rate fck/16
  ldi  r17,(1<<SPE)|(1<<MSTR)|(1<<SPR0)
  out  SPCR,r17
  ret

SPI_MasterTransmit:
  ; Start transmission of data (r16)
  out  SPDR,r16
Wait_Transmit:
  ; Wait for transmission complete
  sbis SPSR,SPIF
  rjmp Wait_Transmit
  ret
```

C Code Example[1]

```
void SPI_MasterInit(void)
{
  /* Set MOSI and SCK output, all others input */
  DDR_SPI = (1<<DD_MOSI)|(1<<DD_SCK);
  /* Enable SPI, Master, set clock rate fck/16 */
  SPCR = (1<<SPE)|(1<<MSTR)|(1<<SPR0);
}

void SPI_MasterTransmit(char cData)
{
  /* Start transmission */
  SPDR = cData;
  /* Wait for transmission complete */
  while(!(SPSR & (1<<SPIF)))
    ;
}
```

Note: 1. See “About Code Examples” on page 7.

19.7.2 CADCSRB - CC-ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0xE5)	–	CADACIE	CADRCIE	CADICIE	–	CADACIF	CADRCIF	CADICIF	CADCSRB
Read/Write	R	R/W	R	R/W	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7 - Res: Reserved**

This bit is reserved bit and will always read as zero.

- **Bit 6 - CADACIE: CC-ADC Accumulate Current Interrupt Enable**

When the CADACIE bit is set (one), and the I-bit in the Status Register is set (one), the CC-ADC Accumulate Current Interrupt is enabled.

- **Bit 5 - CADRCIE: CC-ADC Regular Current Interrupt Enable**

When the CADRCIE bit is set (one), and the I-bit in the Status Register is set (one), the CC-ADC Regular Current Interrupt is enabled.

- **Bit 4 - CADICIE: CC-ADC Instantaneous Current Interrupt Enable**

When the CADICIE bit is set (one), and the I-bit in the Status Register is set (one), the CC-ADC Instantaneous Current Interrupt is enabled.

- **Bits 3 - Res: Reserved**

This bit is reserved bit and will always read as zero.

- **Bit 2 - CADACIF: CC-ADC Accumulate Current Interrupt Flag**

The CADACIF bit is set (one) after the Accumulate Current conversion has completed. The CCADC Accumulate Current Interrupt is executed if the CADACIE bit and the I-bit in SREG are set (one). CADACIF is cleared by hardware when executing the corresponding Interrupt Handling Vector. Alternatively, CADACIF is cleared by writing a logic one to the flag.

- **Bit 1 - CADRCIF: CC-ADC Regular Current Interrupt Flag**

The CADRCIF bit is set (one) when the absolute value of the result of the last CC-ADC conversion is greater than, or equal to, the compare values set by the CC-ADC Regular Current Level Register.

- **Bit 0 - CADICIF: CC-ADC Instantaneous Current Interrupt Flag**

The CADICIF bit is set (one) when a CC-ADC Instantaneous Current conversion is completed. The CC-ADC Instantaneous Current Interrupt is executed if the CADICIE bit and the I-bit in SREG are set (one). CADICIF is cleared by hardware when executing the corresponding Interrupt Handling vector. Alternatively, CADICIF is cleared by writing a logic one to the flag.

19.7.5 CADRC- CC-ADC Regular Current

Bit	7	6	5	4	3	2	1	0	
(0xE6)	CADRC[7:0]								CADRC
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 CC-ADC Regular Current Register determines the threshold level for the Regular Current detection. When the result of a CC-ADC Instantaneous Current conversion has an absolute value greater than, or equal to, the Regular Current level, the CC-ADC Regular Current Interrupt Flag is set.

The value in this register defines the eight least significant bits of the Regular Current level. The most significant bits of the Regular Current level are always zero. The programmable range for the Regular Current level is given in Table 19-3.

Table 19-3. Programmable Range for the Regular Current Level

		Minimum	Maximum	Step size
Voltage (µV)		0	6848	26.9
Current (mA)	R_{SENSE} = 1 mΩ	0	6848	26.9
	R_{SENSE} = 5 mΩ	0	1370	5.4
	R_{SENSE} = 10 mΩ	0	685	2.7

The CC-ADC Regular Current Register does not affect the setting of the CC-ADC Conversion Complete Interrupt Flag.

20.4 Register Description

20.4.1 VADMUX – V-ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	–	–	–	–	VADMUX3	VADMUX2	VADMUX1	VADMUX0	VADMUX
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:4 – RES: Reserved Bits**

These bits are reserved bits in the ATmega8HVA/16HVA and will always read as zero.

- **Bit 3:0 – VADMUX3:0: V-ADC Channel Selection Bits**

The VADMUX bits determine the V-ADC channel selection. See Table 20-1 on page 114.

Table 20-1. VADMUX channel selection

VADMUX3:0	Channel Selected	Scale
0000	RESERVED	–
0001	CELL 1	0.2
0010	CELL 2	0.2
0011	RESERVED	–
0100	VTEMP(1)	1.0
0101	ADC0	1.0
0110	ADC1	1.0
0111...1111	RESERVED	–

Note: 1. VTEMP must be measured in Active mode to get the highest accuracy when using calibration value stored in signature row.

20.4.2 VADCSR – V-ADC Control and Status Register

Bit	7	6	5	4	3	2	1	0	
(0x7A)	–	–	–	–	VADEN	VADSC	VADCCIF	VADCCIE	VADCSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:4 – RES: Reserved Bits**

These bits are reserved bits in the ATmega8HVA/16HVA and will always read as zero.

- **Bit 3 – VADEN: V-ADC Enable**

Writing this bit to one enables V-ADC conversion. By writing it to zero, the V-ADC is turned off. Turning the V-ADC off while a conversion is in progress will terminate this conversion. Note that the bandgap voltage reference must be enabled separately, see “BGCCR – Bandgap Calibration C Register” on page 118.

- **Bit 2 – VADSC: Voltage ADC Start Conversion**

Write this bit to one to start a new conversion of the selected channel.

Figure 22-3. Voltage Regulator block diagram, Linear mode only

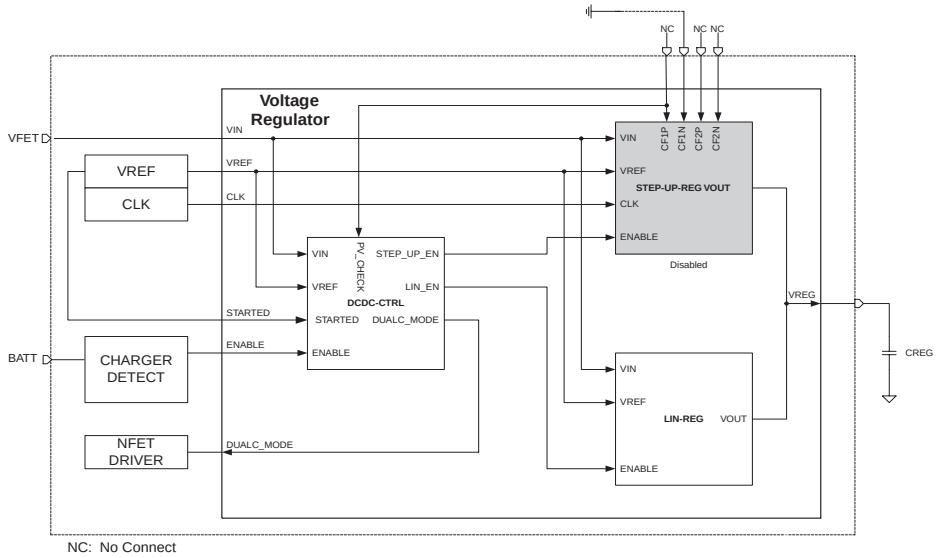
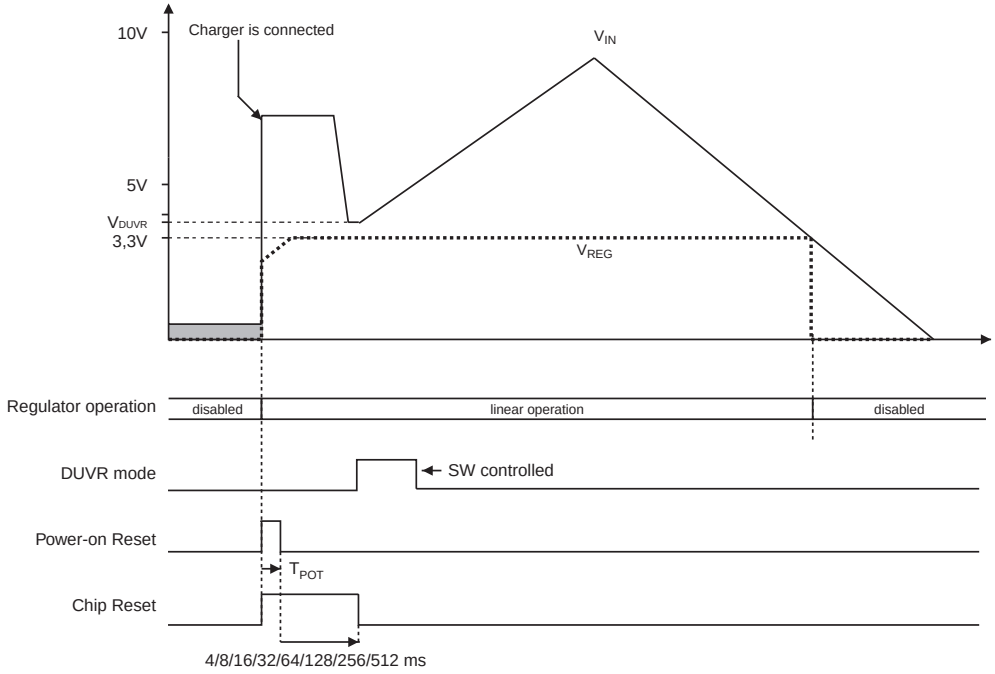


Figure 22-4. Voltage regulator operation and reset signals as a function of rising and falling input voltage for 2-cell operation.



23.9.9 BPDHCD – Battery Protection Discharge-High-current Detection Level Register

Bit	7	6	5	4	3	2	1	0	
(0xF8)	DHCDL[7:0]								BPDHCD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	1	1	1	0	0	1	1	

- **Bits 7:0 – DHCDL7:0: Discharge High-current Detection Level**

These bits sets the R_{SENSE} voltage level for detection of Discharge High-current, as defined in Table 23-5 on page 132.

Note: Due to synchronization of parameters between clock domains, a guard time of 3 ULP oscillator cycles + 3 CPU clock cycles is required between each time the BPDHCD register is written. Any writing to the BPDHCD register during this period will be ignored.

23.9.10 BPCHCD – Battery Protection Charge-High-current Detection Level Register

Bit	7	6	5	4	3	2	1	0	
(0xF9)	CHCDL[7:0]								BPCHCD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	1	1	1	0	0	1	1	

- **Bits 7:0 –CHCDL7:0: Charge High-current Detection Level**

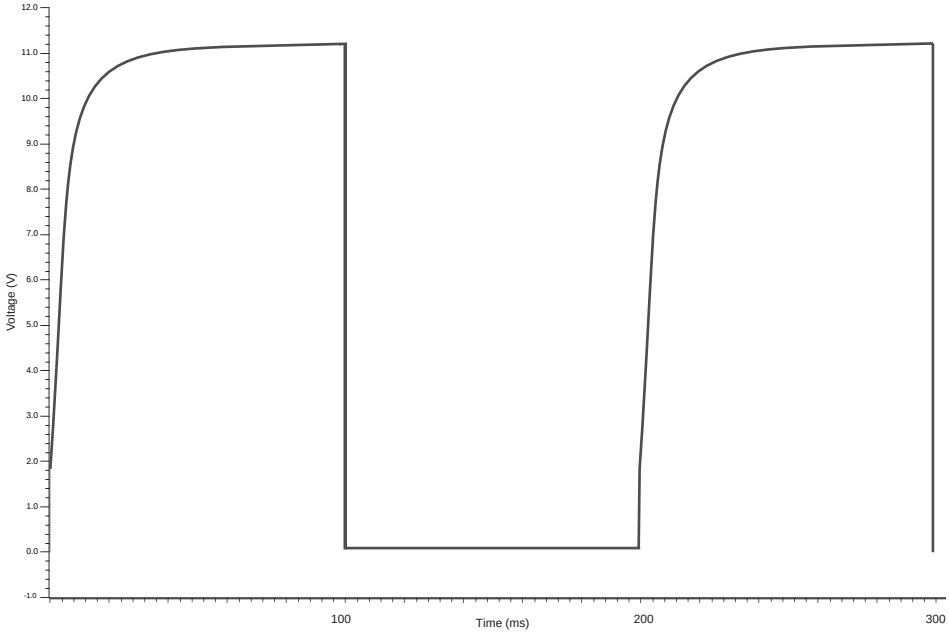
These bits sets the R_{SENSE} voltage level for detection of Charge High-current, as defined in Table 23-5 on page 132.

Note: Due to synchronization of parameters between clock domains, a guard time of 3 ULP oscillator cycles + 3 CPU clock cycles is required between each time the BPCHCD register is written. Any writing to the BPCHCD register during this period will be ignored.

Table 23-5. DL[7:0] with corresponding R_{SENSE} Current for all Current Detection Levels (R_{SENSE} = 10 mΩ, VREF = 1.100 ± 0.005V, T_A = -10°C to 70°C)

Current Protection Detection Level			
DL[7:0]	**Min.**	**Typ.**	**Max.**
0xF3		2.0A	
0xF4		2.5A	
0xF5		3.0A	
0xF6		3.5A	
0xF7		4.0A	
0xF8		4.5A	
0xF9		5.0A	
0xFA		5.5A	
0xFB		6.0A	
0xFC		6.5A	
0xFD		7.0A	

Figure 24-3. Switching NFET on and off during NORMAL operation



24.3 DUVR – Deep Under-Voltage Recovery Mode operation

The purpose of DUVR mode is to control the Charge FET so that the VFET voltage is above the minimum operating voltage while charging cells below minimum operating voltage. This is useful when the cell has been discharged below the minimum operating voltage of the chip. In DUVR mode the Charge FET is switched partly on to provide a suitable voltage drop between the cell voltage and the VFET terminal. As the cell voltage increases, the voltage drop across the Charge FET will gradually decrease until the Charge FET is switched completely on. This means that for high cell voltages, DUVR mode operation is equivalent to normal enabling of the Charge FET (CFE=1).

ATmega8HVA/16HVA should operate in DUVR mode until software detects that the cell has recovered from Deep Under-Voltage condition. When the cell has recovered from Deep Under-Voltage condition, software should first set CFE=1. This is safe now since the cell voltage is above minimum operating voltage. After that software should disable DUVR mode by setting DUVRD = 1.

If both DUVRD and CFE bit is set before the cell voltage is above minimum operating voltage, the VFET voltage will drop and the chip will enter BOD reset and switch off both the Charge- and Discharge FET. Switching off the FET's will cause the VFET voltage to rise again so that the chip restarts from BOD reset, with DUVRD = 0 and CFE = 0 (default values). To avoid this, software must always check the cell voltage by V-ADC measurements before setting CFE=1.

DUVR mode is default enabled after reset. However, while the chip is in reset state, DUVR mode is disabled. This is a safety feature that ensures that the Charge FET will not be switched on until the Charge Over-current Protection is operating. This implies that the DUVR mode will be disabled from the time that a charger is connected until the selected start-up time expired. During this period, the VFET voltage will be higher than the normal VFET Level in DUVR mode.

For more details about DUVR mode, refer to application note AVR354.

26. Self-Programming the Flash

26.1 Overview

The device provides a Self-Programming mechanism for downloading and uploading program code by the MCU itself. The Self-Programming can use any available data interface and associated protocol to read code and write (program) that code into the Program memory.

The Program memory is updated in a page by page fashion. Before programming a page with the data stored in the temporary page buffer, the page must be erased. The temporary page buffer is filled one word at a time using SPM and the buffer can be filled either before the Page Erase command or between a Page Erase and a Page Write operation:

Alternative 1, fill the buffer before a Page Erase

- Fill temporary page buffer
- Perform a Page Erase
- Perform a Page Write

Alternative 2, fill the buffer after Page Erase

- Perform a Page Erase
- Fill temporary page buffer
- Perform a Page Write

If only a part of the page needs to be changed, the rest of the page must be stored (for example in the temporary page buffer) before the erase, and then be re-written. When using alternative 1, the Boot Loader provides an effective Read-Modify-Write feature which allows the user software to first read the page, do the necessary changes, and then write back the modified data. If alternative 2 is used, it is not possible to read the old data while loading since the page is already erased. The temporary page buffer can be accessed in a random sequence. It is essential that the page address used in both the Page Erase and Page Write operation is addressing the same page.

26.1.1 Performing Page Erase by SPM

To execute Page Erase, set up the address in the Z-pointer, write “00000011” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE in the Z-register. Other bits in the Z-pointer will be ignored during this operation.

- The CPU is halted during the Page Erase operation.

26.1.2 Filling the Temporary Buffer (Page Loading)

To write an instruction word, set up the address in the Z-pointer and data in R1:R0, write “00000001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The content of PCWORD in the Z-register is used to address the data in the temporary buffer. The temporary buffer will auto-erase after a Page Write operation or by writing the CTPB bit in SPMCSR. It is also erased after a system reset. Note that it is not possible to write more than one time to each address without erasing the temporary buffer.

If the EEPROM is written in the middle of an SPM Page Load operation, all data loaded will be lost.

26.2.6 Programming Time for Flash when Using SPM

The Fast RC Oscillator is used to time Flash accesses. Table 26-2 shows the typical programming time for Flash accesses from the CPU.

Table 26-2. SPM Programming Time, f_{OSC} = 8.0 MHz[1]

Symbol	Min Programming Time	Max Programming Time
Flash write (Page Erase, Page Write, and write Lock bits by SPM)	3.7 ms	4.5 ms

Note: 1. Minimum and maximum programming times is per individual operation.

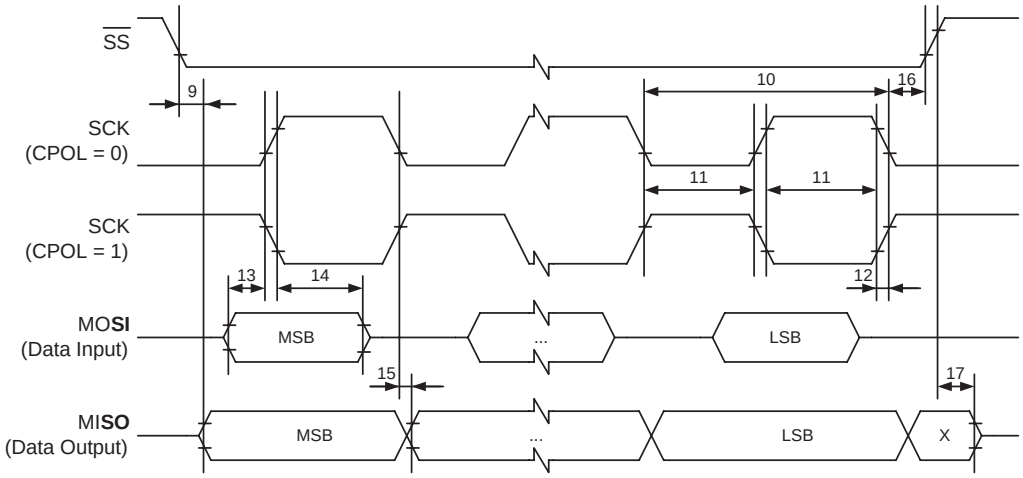
Table 26-3. Explanation of different variables used in Figure 26-1 and the mapping to the Z-pointer, ATmega8HVA.

Variable		Corresponding Z-value	Description
PCMSB	11		Most significant bit in the Program Counter. (The Program Counter is 12 bits PC[11:0])
PAGEMSB	5		Most significant bit which is used to address the words within one page (64 words in a page requires six bits PC [5:0]).
ZPCMSB		Z12	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPCMSB equals PCMSB + 1.
ZPAGEMSB		Z6	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPAGEMSB equals PAGEMSB + 1.
PCPAGE	PC[11:6]	Z12:Z7	Program Counter page address: Page select, for Page Erase and Page Write
PCWORD	PC[5:0]	Z6:Z1	Program Counter word address: Word select, for filling temporary buffer (must be zero during Page Write operation)

Table 26-4. Explanation of different variables used in Figure 26-1 and the mapping to the Z-pointer, ATmega16HVA.

Variable		Corresponding Z-value	Description
PCMSB	12		Most significant bit in the Program Counter. (The Program Counter is 13 bits PC[12:0])

SPI Interface Timing Requirements (Slave Mode)



29.8 Programming Characteristics

29.8.1 Serial Programming

Figure 29-2. Serial Programming Timing

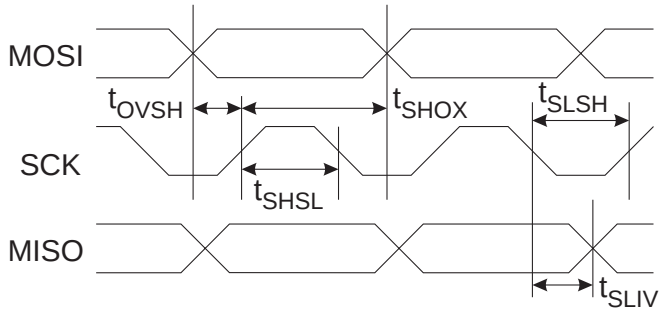


Figure 29-3. Serial Programming Waveforms

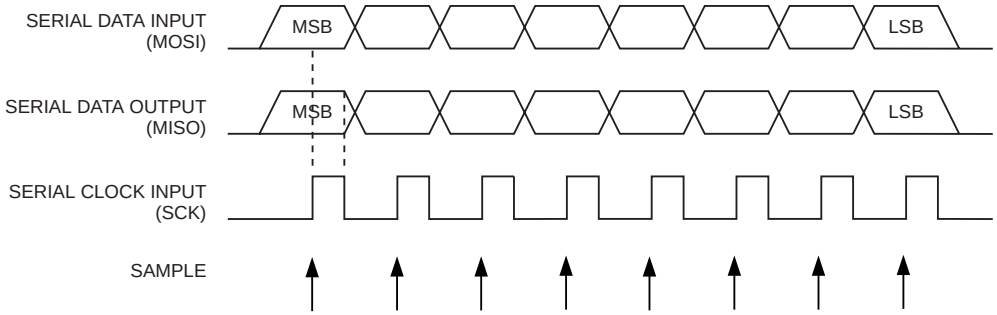


Table of Contents