Atmel AVR042: AVR Hardware Design Considerations

Features

- **Providing robust supply voltage, digital and analog.**
- **Connection of RESET line.**
- **SPI interface for In-System Programming.**
- **Using external crystal or ceramic resonator oscillators.**

1 Introduction

This application note has been written to provide answers to some of the questions and problems faced when starting designs involving Atmel®AVR® microcontrollers. The application note treats topics that are known to cause problems. The scope is to provide an introduction to potential design problems rather than being an exhaustive walk-through of how to design applications using the AVR microcontrollers. This document is thus a collection of information from existing Atmel AVR documents, combined with information that is not previously documented.

It is highly recommended to read the application note AVR040 - "EMC Design Considerations" – before initiating a new design, especially if the aim of the design is a commercial application that needs to meet the requirements of the EMC directive (or similar directives in countries outside Europe). The application note is available from the AVR section of the official Atmel website http://www.atmel.com.

2 Power Supply

Two aspects should be considered when designing the power supply for the discrete/digital elements of an Atmel®AVR®; ESD protection and noise emission. Both these topics are treated in details in the AVR040 application note, and only a short summary is included in this document.

2.1 Digital supply

Looking at the datasheet for an AVR microcontroller, one can be fooled to believe that power supply is not critical. The device has a very wide voltage range, and draws only a few mA supply current. But as with all digital circuits, the supply current is an average value. The current is drawn in very short spikes on the clock edges, and if I/O lines are switching, the spikes will be even higher. The current pulses on the power supply lines can be several hundred mA if all eight I/O lines of an I/O port changes value at the same time. If the I/O lines are not loaded, the pulse will only be a few ns.

This kind of current spike cannot be delivered over long power supply lines; the main source is (or should be) the decoupling capacitor.

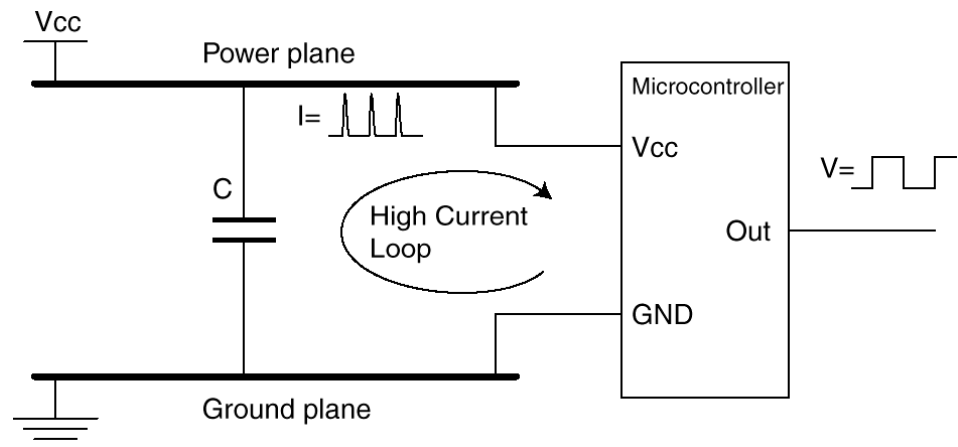**Figure 2-1.** Incorrect decoupling

Figure 2-1 shows an example of insufficient decoupling. The capacitor is placed too far away from the microcontroller, creating a large high current loop. The power and ground planes here are parts of the high current loop. As a result of this, noise is spread more easily to other devices on the board, and radiated emission from the board is increased even further. The whole ground plane will act as an antenna for the noise, instead of only the high current loop. This will be the case if the power and ground pins are connected directly to the planes (typical for hole-mounted components) and the decoupling capacitor is connected the same way. The same is often seen for boards with surface-mount components if the integrated circuits are placed on one side of the board and the decoupling capacitors are placed on the other.

Figure 2-2 shows a better placement of the capacitor. The lines that are part of the high current loop are not part of the power or ground planes. This is important, as the power and ground planes otherwise will spread a lot of noise. Further, the figure

shows another improvement of the decoupling. A series ferrite bead is inserted to reduce the switching noise on the power plane. The series resistance of the ferrite bead must of course be low enough to ensure that there will be no significant DC voltage drop.

Figure 2-2. Decoupling with series inductor.
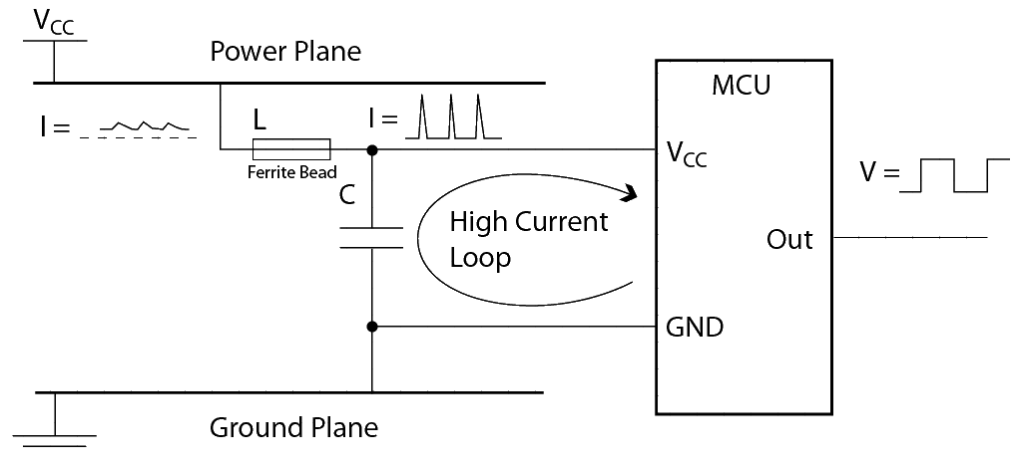
Generally, the Atmel®AVR® devices where power and ground lines are placed close together (like the ATmega8535) will get better decoupling than devices with industry standard pin-out (like the ATmega8515), where the power and ground pins are placed in opposite corners of the DIP package. This disadvantage can be overcome by using for example a TQFP package, which allows decoupling capacitors to be placed very close to the die. For devices with multiple pairs of power and ground pins, it is essential that every pair of pins get its own decoupling capacitor.

The main supply should also have a tantalum or ceramic capacitor of some μF to stabilize it.

2.2 Analog supply

The AVR devices that have built-in ADC have a separate analog supply voltage pin, AVcc. This separate voltage supply is provided to make the analog circuits less prone to the digital noise originating from the switching of the digital circuits.

To be able to obtain good accuracy with the ADC the analog supply voltage must be decoupled separately, in the same manner as the digital supply voltage. AREF must also be decoupled, a typical value for the capacitor is 100nF. If a separate analog ground (AGND) is present, the analog ground should be separated from the digital ground, - so that the analog and digital grounds are only connected at one point, - the origin of the GND i.e. at the power supply GND.

3 Connection of RESET pin on AVRs

The RESET pin on the AVR is active LOW, and setting the pin LOW externally will thus result in a reset of the AVR. The RESET has two purposes:

1. To release all lines by tri-stating all pins (except XTAL pins), initialize all I/O registers and set program counter to zero.

2. To enter programming mode (for some parts also the PEN line is used to enter programming mode). Furthermore it is possible to enter high-voltage/parallel programming mode by drawing the RESET pin "very" high, where very high means 11.5 – 12.5V (refer to the datasheet of the device for more information).

The reset line has an internal pull-up resistor, but if the environment is noisy it can be insufficient and reset can therefore occur sporadically. Refer to datasheet for value of pull-up resistor on specific devices.

Connecting the RESET so that it is possible to enter both high-voltage programming and ordinary low level reset can be achieved by applying a pull-up resistor to the RESET line. This pull-up resistor makes sure that reset does not go low unintended. The pull-up resistor can in theory be of any size, but if the Atmel®AVR® should be programmed from e.g. STK500/AVRISP the pull-up should not be so strong that the programmer cannot activate RESET by draw the RESET line low. The recommended pull-up resistor is 4.7kΩ or larger when using STK500 for programming. For debugWIRE to function properly, the pull-up must not be smaller than 10kΩ.

To protect the RESET line further from noise, it is an advantage to connect a capacitor from the RESET pin to ground. This is not directly required since the AVR internally have a low-pass filter to eliminate spikes and noise that could cause reset. Applying an extra capacitor is thus an additional protection. However, note that this capacitor cannot be present if debugWIRE or PDI is used.

If not using High Voltage Programming it is recommended to add an ESD protecting diode from RESET to Vcc, since this is not internally provided due to High Voltage Programming. Alternatively, or in addition, a zener diode can be used to limit the RESET voltage relative to GND. The zener diode is highly recommended in noisy environments. The components should be located physically close to the RESET pin of the AVR. Figure 3-1 shows the recommended circuit on the RESET line.

3.1 External RESET switch

If an external switch is connected to the RESET pin it is important to add a series resistance. Whenever the switch is pressed it will short the capacitor, the current through the switch can have high peak values. This will cause the switch to bounce and give steep spikes in 2-10ms periods until the capacitor is discharged. The PCB tracks and the switch metal will introduce a small inductance, this inductance can generate high voltages according to $Vi = L* di/dt$. These voltages are most likely outside the specification of the RESET pin. By adding a series resistor between the switch and the capacitor the peak currents generated will be significantly, these currents won't be large enough to generate high voltages at the RESET pin. An example connection is shown in Figure 3-2.

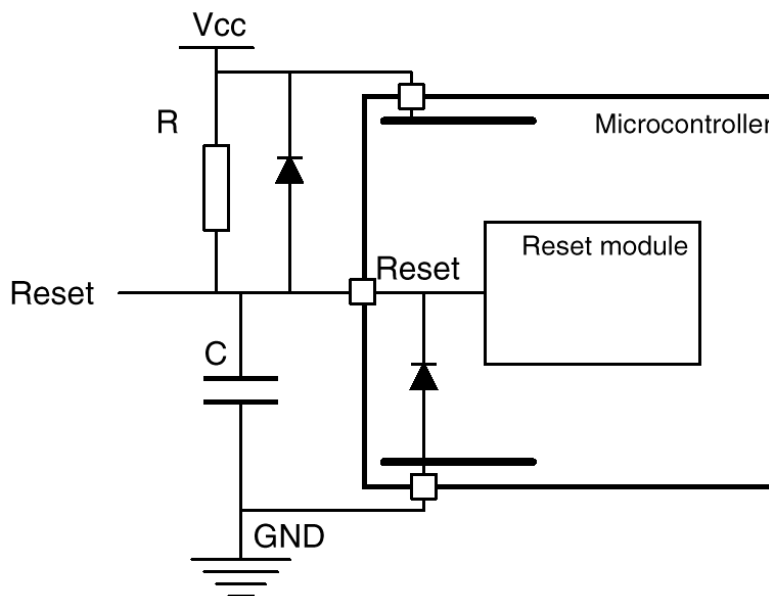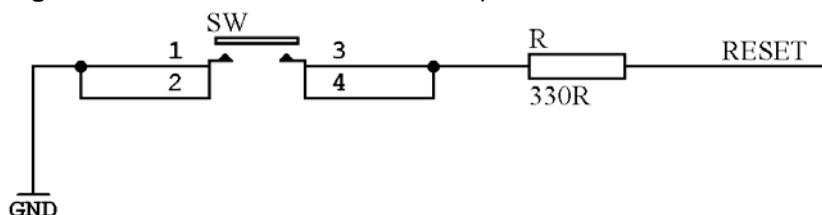**Figure 3-1.** Recommended Reset Pin connection.

Figure 3-2 Switch connection for Reset pin.

4 Connecting ISP lines

Atmel®AVR® microcontrollers feature one or more interfaces for In-System Programming (ISP). These are used for programming the Flash, EEPROM, Lock-bits and most Fuse-bits in all AVRs (except the ATtiny11 and ATtiny28). This feature makes it possible to program the AVR in the last stage of production of a target application board, reprogram if SW bugs are identified late in the process, or even update the AVR in the field if required. Some of the ISP interfaces may also be used for on-chip debugging. It is therefore highly recommended to always design the target application board so that the ISP connectors can be accessed in some way — just in case.

The different programming interfaces are introduced in the following subsections. Refer to the device datasheet and AVR Tools help for further information on which interfaces are supported by the device, and how to connect the programming tool.

4.1 SPI programming interface

On devices that use a Serial Peripheral Interface (SPI) for ISP, these lines are usually located on the same pins as regular SPI, or else on pins that can be used for other purposes. Consult the device data sheet to find out which pins are used for ISP.

Two standard SPI connectors are provided by the Atmel ISP programmers; a 6 pin and a 10 pin connector. These are seen in Figure 4-1. In addition to the data lines

(MOSI and MISO) and the bus clock (SCK), target voltage VTG, GND and RESET (RST) are provided through these connectors.

**Figure 4-1.** Standard ISP connectors on STK500, AVR ISP and STK200/STK300

Some ISP programmers are powered by the target power supply. In this way they easily adapt to the correct voltage level of the target board. Other ISP programmers, like STK500, can alternatively power the target board via the VTG line. In that case it is important that the power supply on the target is not switched on. Read the User Guide for your ISP programmer for details on its capabilities and physical interface

4.1.1 Shared use of SPI programming lines

If additional devices are connected to the ISP lines, the programmer must be protected from any device, other than the AVR, that may try to drive the lines. This is especially important with the SPI bus, as it is similar to the ISP interface. Applying series resistors on the SPI lines, as depicted in Figure 4-2, is the easiest way to achieve this.

Note that the AVR will never drive the SPI lines in a programming situation, since the AVR is held in RESET to enter programming mode – and RESET'ing the AVR tri-states all pins.

Figure 4-2. Connecting the SPI lines to the ISP interface.

AVR042
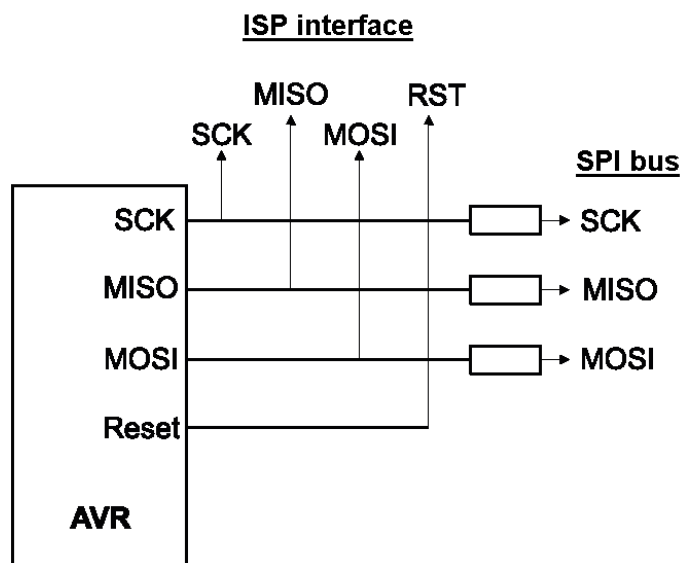
Multiple Atmel®AVR® in a single application can share the same ISP interface to allow for programming of all the devices through a minimal interface. However, the AVR devices will all respond to the ISP instructions if special design considerations are not made.

If it is desired to have only one ISP interface on the target board, the ISP programming can be designed so that only one of the AVR devices is provided with a SPI clock at a time. All other SPI lines can then be shared. This way, several AVRs can be located "behind" the same protection resistors, since they are all held in RESET while the ISP reset line is activated. The gating of the ISP clock can be accomplished by use of, e.g., jumpers or DIP switches. An alternate solution is to have multiple ISP interfaces, one for each device, all protected as shown in Figure 4-2.

4.2 JTAG interface

Some devices have a JTAG interface, which can be used for both programming and debugging. The JTAG lines are shared with analog input and should be connected so that the JTAG programmer can get control of the lines. Though JTAG programming tools such as the JTAGICE mkII® can drive a resistive load (refer to AVR Studio help for details), capacitive load should in general be avoided.

Figure 4-3 shows the standard JTAG connector supplied with Atmel ISP programmers. Just as for the SPI programming connector, the target's voltage supply is made available to allow for powering of the device or ensuring correct signal levels when programming.
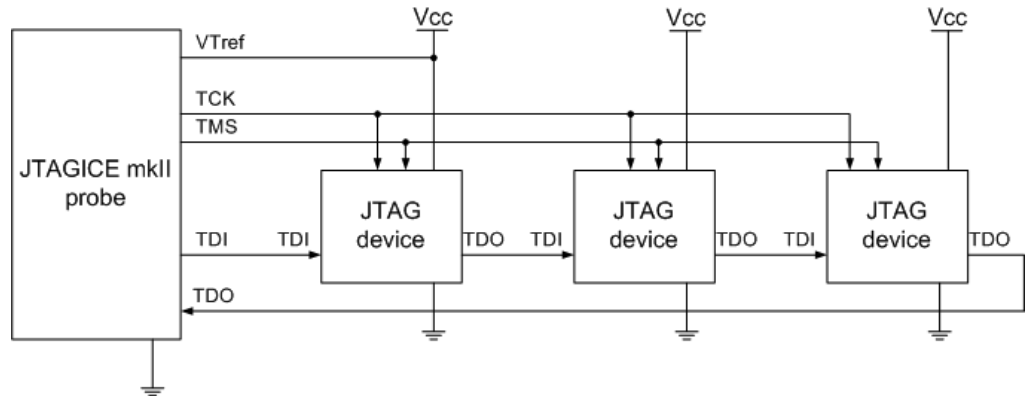
Figure 4-3. Standard JTAG connector used on e.g. JTAGICE mkII and AVR ONE!
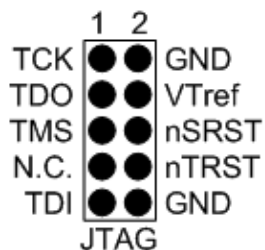

4.2.1 Shared use of JTAG lines

By creating a JTAG daisy-chain, a single JTAG connector can serve as an ISP interface for several devices. This principle is shown in Figure 4-4 and is explained in further detail in the AVR Tools help file.

Figure 4-4: JTAG daisy-chain.

Protection resistors, as shown in Figure 4-2, are necessary if the JTAG lines are also used for other purposes. For example, if they are used for analog input to the Atmel®AVR®, there are often analog filters on the lines. If such filters are present on the JTAG lines, it may be required to eliminate the filter capacitor during programming to ensure that the load is mainly resistive. Figure 4-5 illustrates how this can be done.

Figure 4-5. Filters on JTAG pins – correct and incorrect ways to connect.

AVR042
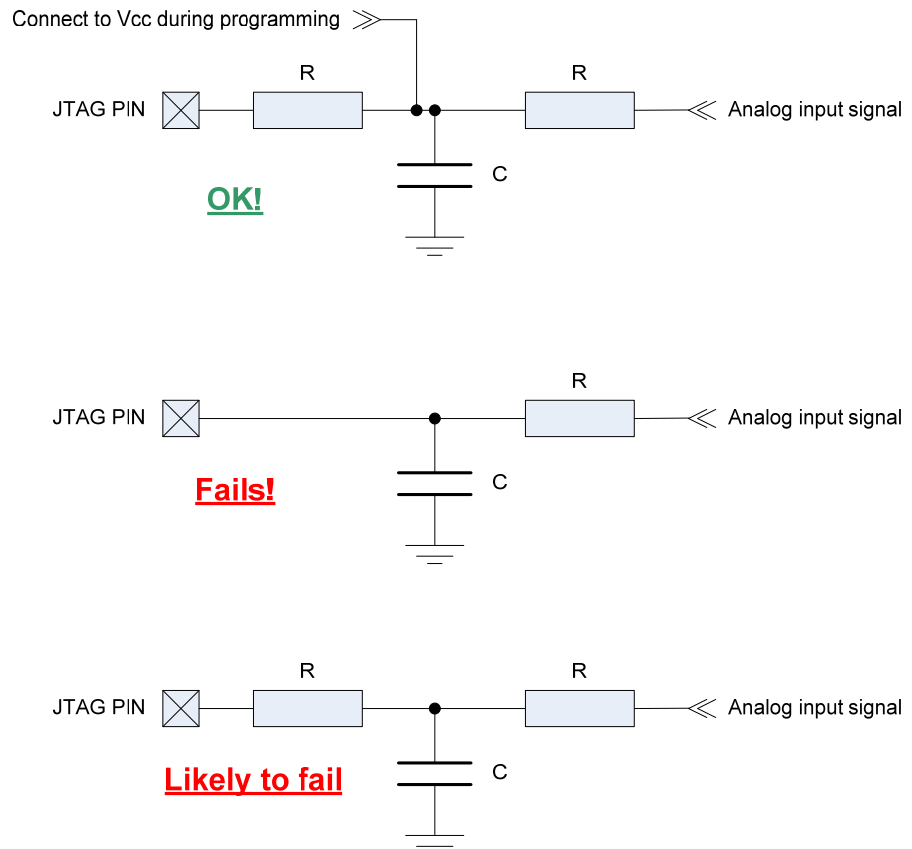
4.3 PDI interface

The Program and Debug Interface (PDI) is an Atmel proprietary two-line interface that was introduced with the AVR XMEGA microcontroller family. As the name implies, this interface can be used for both In-System Programming and on-chip debugging of devices.

Figure 4-6 shows the standard PDI connector supplied with Atmel programmers. Only two pins on the device are needed for use of this interface: RESET, also called PDI_CLK, and the dedicated PDI_DATA pin. The target's voltage supply is made available to allow for powering of the device or ensuring correct signal levels during programming.

Figure 4-6: Standard PDI connector used on e.g. STK600 and JTAGICE mkII.
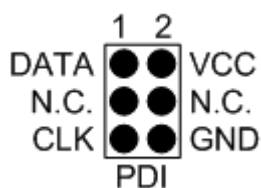
4.3.1 External reset circuitry

Since the RESET line is used for clocking the PDI, it is important to bypass or avoid any circuitry that can distort the clock signal during programming or debugging, such as capacitors and external reset sources. During normal operation the RESET pin has an internal filter to prevent unintentional resets, e.g. caused by short spikes on the reset line. Despite the fact that the clock signal is deformed capacitive loads up to 1nF have been tested to work with the STK600, AVRISP mkII and AVR Dragon during programming and the JTAGICE mkII® and the AVR ONE! during programming and debugging.

Pull-up resistors should be at least 10kΩ, or be removed from the RESET line if an Atmel programmer is used.

4.4 TPI interface

The Tiny Programming Interface is featured on the Atmel® tinyAVR® devices with the lowest pin count.

Figure 4-7 shows the standard TPI connector supplied with Atmel programmers. Only three pins on the device are needed for use of this interface: RESET, TPICLK and TPIDATA. The two latter pins are multiplexed with regular I/O pins.

Figure 4-7: Standard TPI connector used on e.g. STK600 and AVRISP mkII.
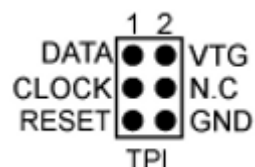
4.4.1 Disabling RESET pin functionality

The RESET pin can be reconfigured as an I/O pin by programming the RSTDISBL fuse of the device. This disables the reset functionality, and requires +12V to be applied to RESET for programming to work. Not all programming tools are capable of generating this voltage.

5 Using crystal and ceramic resonators

Most Atmel®AVR® MCUs can use different clock sources. The optional external clock sources are Clock, RC oscillator, crystal or ceramic resonator. The use of crystals and ceramic resonators are in some designs causing problems due to the fact that the use of these clock sources is not well understood. This section therefore treats the topic of using crystals and ceramic resonators in relation to AVR MCUs. The description focus on features and parameters relevant for designing applications where crystals or ceramic resonators are used rather than trying to be a complete description of the theory related to the topic. For more information and theory regarding crystals please refer to application note "AVR4100: Selecting and testing 32kHz crystal oscillators for AVR microcontrollers"

5.1 Selecting the clock source in the AVR

The clock source used by the AVR is selected through the fuses for all parts except the Xmega family where the clock source is selected by software. Most ISP and parallel programmers can program the fuses that are related to selecting a clock source. The fuses are not erased when the AVR memory is erased and the fuses therefore only require to be programmed if the fuse settings should be altered. Programming the fuses each time the device is erased and reprogrammed is thus not necessary. The clock options that are relevant for this document are:

- "Ext. low-frequency crystal"
- "Ext. crystal oscillator"
- "Ext. ceramic resonator"

Several sub-settings relating to the start-up time of the AVR can be selected, but the 3 clock options mentioned are the fundamental settings that should be focused on. The clock options available can vary between AVR devices, - not all devices have the ability to run on various external oscillators. Check the datasheet for the relevant device to determine the clock options.

One should be aware that if selecting a different clock source than is actually applied, the AVR might not run since different oscillator circuits are activated internally in the AVR dependent on the selected clock option. Since the fuses are not cleared by a memory erase, it can cause problems if incorrect settings are chosen.

5.2 General about crystals and ceramic resonators

The typical type of crystal used for the AVR is the AT-cut parallel resonant crystal. The ceramic resonator is very similar to the AT-cut parallel resonant crystal, but is so to say a low cost, low quality version of the crystal. The ceramic resonator has a lower Q-value, which is both an advantage and disadvantage. Due to the lower Q-value the oscillator frequency of the ceramic resonator can more easily be "tuned" to

a desired frequency, but is also more sensitive to temperature and load changes, causing undesired frequency variations. The advantage of the ceramic resonators is that it has a faster start-up than crystals.

In general there will not be distinguished between crystals and ceramic resonators in this section and the term "resonator" will thus refer to both devices.

Table 5-1. Technical differences between ceramic resonators and quarts crystal

	Ceramic resonator	Quartz crystal
Aging	±3000ppm	± 10ppm
Frequency tolerance	± 2000-5000ppm	± 20ppm
Freq. temperature characteristics	± 20-50ppm/˚C	± 0.5ppm/˚C
Frequency pullability	± 100-350ppm/pF	± 15ppm/pF
Oscillator rise time	0.01-0.5ms	1-10ms
Quality factor (Qm)	100-5000	10^3-$5 \cdot 10^5$

The parallel resonator is intended used in circuits, which contains reactive components, usual capacitors. Such circuits depend on the combination of the reactive components and the resonator to accomplish the phase shift necessary to start and maintain the oscillation at a given frequency. Basic oscillator circuits used for parallel resonators are illustrated in Figure 5-1. The part of the circuit above the dashed line, represent the oscillator circuit internally in the AVR.

Figure 5-1. Basic inverter circuits equivivalent to the oscillator circuits in AVRs.

(A)   (B)
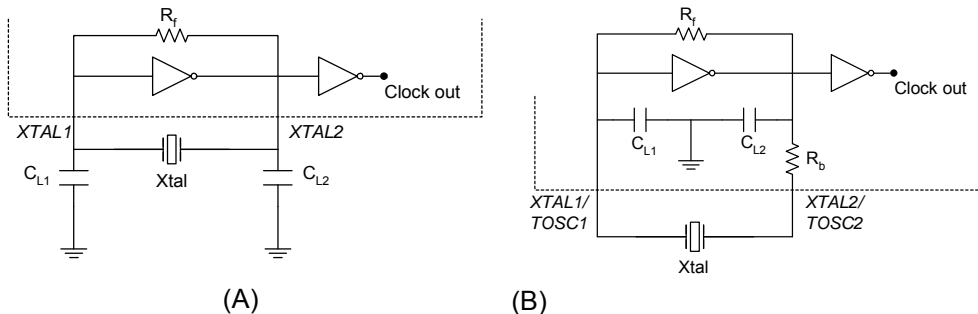
Notes: i.   Oscillator circuit for crystals and ceramic resonators faster than 400kHz
        ii.   Circuit for low frequency crystals (32.768kHz) (not on all Atmel®AVR®)

Simplifying the description of the Atmel®AVR® built-in oscillator circuits they can be understood as the inverter based oscillator circuits illustrated in Figure 5-1. The circuit used with resonators of frequencies *above* 400kHz is depicted in (A). When using this circuit, capacitive load must be applied externally. The oscillator circuit seen in (B) is the circuit used for low frequency crystals on some AVRs - to be more specific - optimized for 32.768kHz crystals. This circuit provides the capacitive load required by the crystal internally and further adds the resistor R_b to bias the crystal and limit the drive current into the crystal. The resistor R_f is, when using CMOS inverters, approximately 1MΩ, and provides a feedback to bias the inverter to operate it in its linear region. Consult data sheet for the relevant device to see if is has internal circuitry for low frequency crystals.

When using resonators with the Atmel®AVR®, it is necessary to apply (external) capacitors according to the requirements of the facilitated resonator. A parallel resonator will not be able to oscillate stable if the capacitive load applied is insufficient. If the capacitive load is too high the oscillation may have problems starting due to *drive level dependency* of the load. The trick is therefore to find an appropriate value for the capacitive load. The value to look for in the data sheet of the crystal is C_L, the recommended capacitive load of the resonator (viewed from the terminals of the resonator). The capacitive load, C_L of the oscillator circuit, including stray capacitances and the capacitances of the XTAL pins of the AVR can be determined empirically or it can be estimated by Equation 5-1.

Equation 5-1.

$$C_L = \frac{C'_{L1} \cdot C'_{L2}}{C'_{L1} + C'_{L2}} \quad , \qquad \begin{aligned} C'_{L1} &= C_{L1} + C_{L1S} \\ C'_{L2} &= C_{L2} + C_{L2S} \end{aligned}$$

Where C_{L1} and C_{L2} refer to the external capacitors seen in Figure 5-1 and C_{L1S} and C_{L2S} are stray capacitances at the XTAL pins of the AVR. Assuming symmetric layout, so that $C_{L1} = C_{L2} = C$ and $C_{L1S} = C_{L2S} = C_S$, then the external capacitors can be determined by Equation 5-2 (C_S can be estimated to be 5-10pF):

Equation 5-2.
$$C = 2 \cdot C_L - C_S$$

5.3 Recommended capacitor values

The recommendations here will work well in most applications, but there is no way to provide general values for the external capacitors that can be guaranteed to work with all resonators.

When using the clock option "ext. crystal oscillator", crystals with a nominal frequency from 400kHz and up can be used. For these standard "high" frequency crystals the recommended capacitor value is in the range 22-33pF.

The clock option "ext. low frequency crystal" is intended for 32.768kHz crystals. When selecting this clock source the internal oscillator circuit might provide the required capacitive load. By programming the CKOPT Fuse, the user can enable internal capacitors on XTAL1 and XTAL2. The value of the internal capacitors is typical 20pF, but can vary. If using a 32.768kHz crystal that does not require more load than this, external capacitors can be left out. Otherwise external capacitive load must be added. In this case the capacitive load value, given by the manufacturer of the crystal, should be used. Then the value of the external capacitors can be determined by Equation 2. The CKOPT Fuse should not be programmed when using external capacitors.

Please reefer to the datasheet to assure whether the device has internal capacitors or not. Note that AT90S8535, Atmega163 and Atmega103 do not have the CKOPT-fuse; instead they have dedicated pins (TOSC1-TOSC2), to connect the 32.768kHz watch crystal to.

Using the clock option that selects "ext. ceramic resonator" it is strongly recommended to consult the datasheet to determine the capacitors to apply. Always use the capacitive load recommended there since the resonant frequency of the ceramic resonators is very sensitive to capacitive load.

5.4 Unbalanced external capacitors

In noisy environments the oscillator can be affected crucially by the noise. If the noise is strong enough the oscillator can "lock up" and stop oscillating. To make the oscillator less sensitive to noise the size of the capacitor at the high impedance input of the oscillator circuit, XTAL1 can be increased slightly. Increasing only one of the capacitors does not affect the total capacitive load much, but unbalanced capacitors can affect the resonant frequency to a higher degree than the change of the total capacitive load. However, unbalanced capacitive loads will affect the duty cycle of the oscillation and therefore one should in general not use unbalanced capacitive loads. This is especially critical if running the Atmel®AVR® close to its maximum speed limit.

5.5 RTC crystals

Many AVR devices have the possibility use asynchronous clocking of the built-in timer/counter. The counter can through this feature be used for real time functions. A 32.768kHz crystal should then be connected to the TOSCx pins of the AVR.

In some AVRs the internal oscillator circuit used with the real time counter provides a capacitive load of approximately 20pF, which should be appropriate for common 32.768kHz crystals. Refer to the data sheet for the relevant device for info about capacitors/size. External capacitors can be applied if the internal load is insufficient for the applied crystal.

5.6 PCB layout

Finally, the importance of the physical location of the resonator in relation to the AVR should be stressed. Always place the resonator as close to the AVR as possible and shield the resonator by surrounding it with a ground plane.

6 Example layout

Figure 6-1 (A-C) shows a schematic and PCB layout using a crystal oscillator for ATxmega32A4; This example assumes a multilayer design with supply and ground plane on separate layers. Decoupling of all digital supply pairs from VCC and isolating AVCC from VCC is seen. Note the short distance between the crystal/capacitors and the ATxmega32A4, the ground plane surrounding the crystal and how close the vias that connect to the planes are to the ATxmega32A4 pins in the layout.

Figure 6-1. (A) Basic schematic of required/recommended connections for ATxmega32A4. (B) Copper PCB layout and (C) top silk prints.

(A)

(B)

(C)