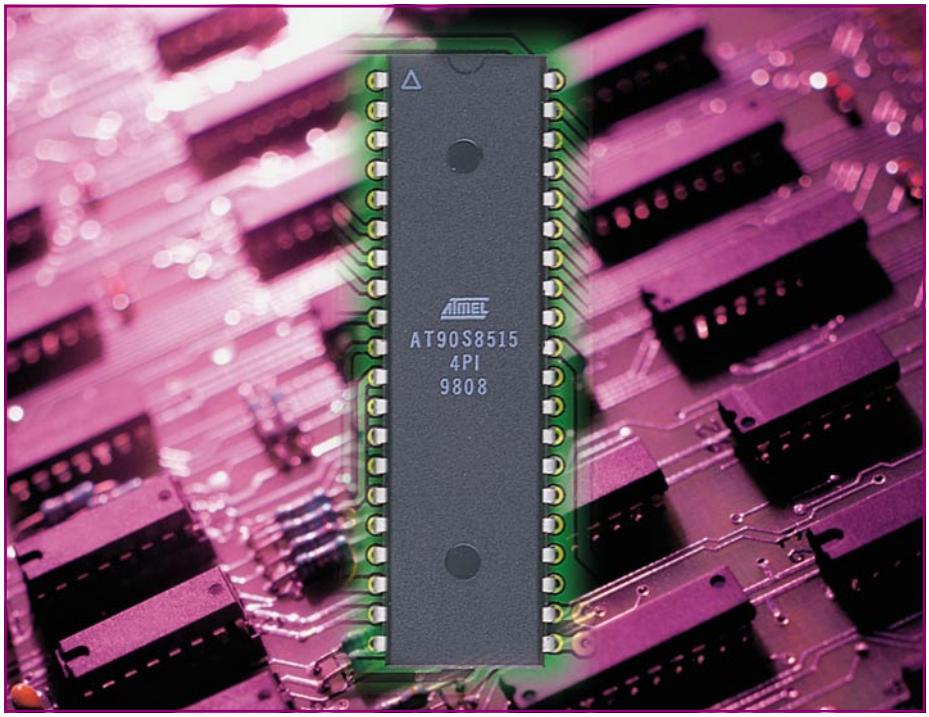

AVR-Grundlagen Teil 1

Die AVR-Mikrocontroller erfreuen sich stetig wachsender Beliebtheit unter Schaltungsentwicklern sowohl im professionellen als auch im privaten Bereich. Sie zeichnen sich durch eine hohe Verarbeitungsgeschwindigkeit und eine Vielzahl verschiedener Typen aus, sodass man quasi zu jedem Projekt über den passenden Controller verfügen kann. In dieser Artikelserie wird die Controller-Baureihe vorgestellt und anhand von Anwendungsbeispielen erläutert.
Der erste Teil befasst sich mit dem Aufbau und der grundlegenden Beschaltung der Controller.

Allgemeines

Mikrocontroller sind heutzutage in fast jedem elektronischen Gerät zu finden, dabei wird oftmals eine hohe Verarbeitungsgeschwindigkeit gefordert, um die zu erledigenden Aufgaben sicher abzuarbeiten. Des Weiteren spielt oft die Ausstattung der Controller in Bezug auf ein internes EEPROM, Analog-Digital-Umsetzer, UART usw. eine große Rolle, weil hierdurch Teile der externen Hardware eingespart werden können. Es werden Kosten eingespart und die Baugruppen fallen kompakter aus.

Die AVR-Mikrocontroller der Firma Atmel weisen zahlreiche dieser Vorteile auf. Sie werden in dieser Artikelserie anhand von Erklärungen und Beispielen vorgestellt. Diese, auch unter Hobby-Elektronikern, immer beliebter werdenden Controller weisen eine große Anzahl beeindruckender Kenndaten auf. Außerdem gibt es eine Vielzahl verschiedener Typen, sodass es für jedes Projekt den passenden Mikrocontroller gibt. Das Konzept dieser Mikrocontroller ist auf eine hohe Performance sowie auf einen geringen Stromverbrauch hin optimiert.

Alle Bausteine der AVR-Serie weisen eine RISC-Architektur (Reduced Instruction Set Computer) auf, bei der fast jeder Befehl in nur einem Taktzyklus abgearbeitet wird. Die Mikrocontroller der MCS51-Familie benötigen im Vergleich dazu für die Abarbeitung eines Befehls 12 Taktzyklen. Das bedeutet, dass ein AVR-Controller, der mit einer Taktfrequenz von 12 MHz arbeitet, bis zu 12 Millionen Befehle pro Sekunde (12 MIPS = Million Instructions Per Second) bewältigen kann, ein MCS51-Controller im Gegensatz dazu aber nur bis zu 1 MIPS.

Da die Programmiersprache „C" in der heutigen Zeit die meistgenutzte Hochsprache in der Mikrocontrollerwelt ist, hat Atmel bereits bei der Konzeption und Entwicklung der AVR-Controller eng mit Experten aus der C-Compiler-Entwicklung zusammengearbeitet und die Architektur und den Befehlssatz so ausgelegt, dass entsprechende C-Programme sehr effizient in den AVR-Maschinencode übersetzt werden können.

Architektur

Ein Mikrocontroller ist ein Mikroprozessor mit Peripheriebeschaltung (E/A-Ports, Timer, UART usw.) auf einem Chip, also ein Ein-Chip-Mikrocomputer. Bei solchen Systemen gibt es unterschiedliche Ansätze bei der Konzeption der Architektur.

Bei der ersten Möglichkeit spricht man von der „Von-Neumann-Architektur", die in vielen modernen Mikrocontrollern ihre Anwendung findet. Hierbei gibt es nur ein Bussystem, auf dem Daten und Befehle übertragen werden können.

Die andere Verfahrensweise wird als „Harvard-Architektur" bezeichnet, bei der es getrennte Bussysteme für Befehle und Daten gibt. Dieses kann, im Vergleich zur vorgenannten Realisierung, die Verarbeitungsgeschwindigkeit der internen Prozesse enorm erhöhen.

Die AVR-Mikrocontroller sind nach dem Harvard-Prinzip aufgebaut, wobei der Datenbus in einer Breite von 8 Bit und der für Befehle als 16 Bit breiter Bus ausgeführt ist. In Abbildung 1 ist das Blockschaltbild der AVR-Architektur zu sehen.

Die ALU (Arithmetic-Logic-Unit) ist das Rechenwerk des Mikrocontrollers. Operationen zwischen Registern werden von der ALU innerhalb eines einzigen Taktzyklus abgearbeitet. Der Registerbereich umfasst 32 Byte, wobei alle Register auch über eine Adresse im Datenspeicher angesprochen werden können.

Der Datenspeicher ist als statisches RAM (SRAM) ausgeführt und umfasst, abhängig vom verwendeten Prozessor-Typ, bis zu 4 KByte. Wie der Name Datenspeicher schon aussagt, können hier alle notwendigen Informationen während des Programmablaufs abgelegt und wieder aufgerufen werden. Die ersten 96 Adressen sind den Arbeitsregistern und den Registern für die Ein- und Ausgabe zugeordnet.

Der nachfolgende Bereich des Datenspeichers beginnt ab der Speicheradresse 60hex. Der Datenspeicher kann sowohl direkt als auch indirekt adressiert werden, was auch aus Abbildung 1 hervorgeht. Außerdem wird dieser Speicherbereich für den Stack genutzt.

Eine Ausnahme bildet hierbei der AT90S1200 und die Controller ATtiny10/11/12/15, welche nur über die 32 Arbeits-

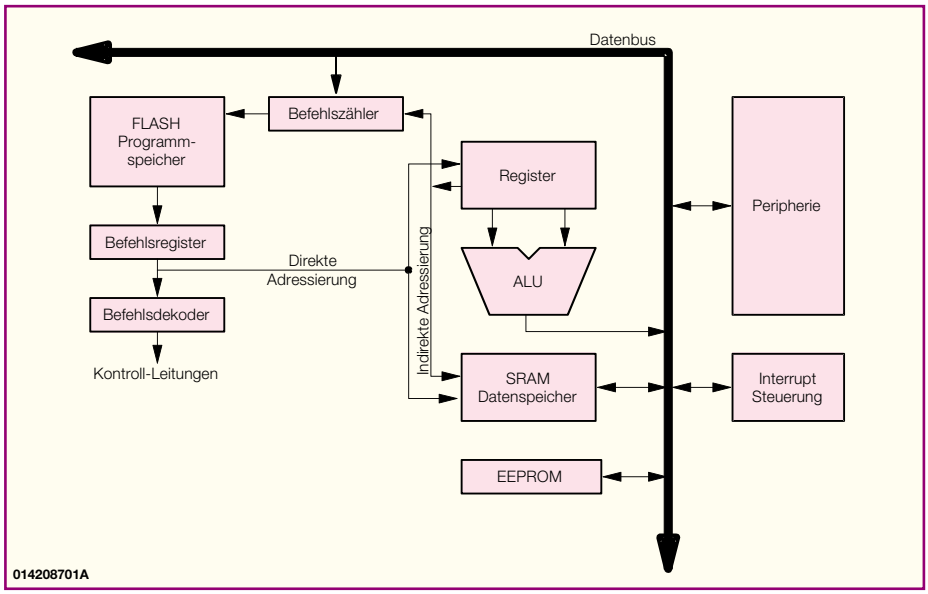

Bild 1: Blockschaltbild der AVR-Architektur

register zur Speicherung der Daten verfügen und bei dem der Stack in Hardware ausgeführt ist.

Zur Speicherung des auszuführenden Programms ist ein FLASH-Speicher integriert, der bis zu 1000-mal beschrieben und wieder gelöscht werden kann. Dieser Programmspeicher kann auch innerhalb der Applikationsschaltung programmiert werden, in der Fachliteratur sowie in Datenblättern findet man diese Funktion häufig unter dem Kürzel „ISP" (In-System-Programmable). Die Größe dieses Speichers ist abhängig vom verwendeten Mikrocontroller. Da alle vorhandenen Befehle 16 Bit breit sind, ist der Bereich des Programmspeichers in 16-Bit-Worten organisiert, d. h., ein Programmspeicher der Größe 2 KByte kann 1 K Befehle speichern.

Des Weiteren sind alle Typen der AVR-Serie mit einem internen EEPROM ausgestattet. Dieses ist als eigener Speicherbereich organisiert, in dem ein byteweises Lesen oder Schreiben über spezielle Kontrollregister erfolgen kann. Das EEPROM hat eine Lebensdauer von bis zu 100.000 Lese-Schreib-Zyklen. Es dient zur Aufnahme von Daten, die auch bei einem Abschalten der Versorgungsspannung noch erhalten bleiben müssen, wie Kalibrierwerte usw.

In Abbildung 2 ist die komplette Speicherübersicht eines AVR-Mikrocontrollers zu sehen.

Ein weiterer wichtiger Teil eines Mikrocontrollers ist der Befehlszähler, der die Adressierung des Programmspeichers vornimmt. Er wird nach jedem Befehl entsprechend erhöht oder, im Falle eines Sprunges im Programm, mit der neuen Adresse geladen. Der adressierte Befehl wird direkt aus dem Programmspeicher in das Befehlsregister geladen, das über den Befehlsdecoder ausgewertet wird.

Die im Blockschaltbild angegebene Peripherie ist auf dem Chip integriert und macht den flexiblen Einsatz eines Mikrocontrollers überhaupt erst möglich. Sie umfasst, abhängig vom Typ, mehr oder weniger Funktionen. Hierzu gehören als wichtigstes Element die Ein-Ausgabeports zur Kommunikation mit der „Außenwelt", über die das entsprechende elektronische Gerät, in dem der Mikrocontroller eingesetzt ist, gesteuert wird. Um für die zu erledigenden Aufgaben immer das richtige Timing zu haben, sind die Timer/Counter, die als 8- oder 16-Bit-Versionen vorhanden sind, das richtige Werkzeug. Außerdem sind für viele Anwendungen Analog-Digital-Umsetzer, PWM-Ausgänge, serielle Schnittstellen oder Komparatoren wichtige Elemente. Wird ein vorhandener Watchdog-Timer („Wachhund") richtig eingesetzt, kann der Programmablauf im Controller überwacht werden. Der Watchdog-Timer löst im Falle einer ungewollten Endlosschleife einen Reset aus und startet das Programm somit von vorn.

Die vorhandene Interrupt-Steuereinheit kann den Programmablauf auf Anforderung, z. B. Überlauf eines Timers, Auftreten einer Flanke am Eingang usw., unterbrechen und führt dann eine entsprechend programmierte Routine aus. Im deutschen Sprachgebrauch wird ein Interrupt auch Unterbrechungsanforderung genannt.

Eine Übersicht diverser AVR-Mikrocontroller mit den zugehörigen Angaben zur Ausstattung kann Tabelle 1 entnommen werden. Das komplette Programm aller Typen dieser Familie ist auf der Atmel-Hompage zu finden.

Beschaltung der wichtigsten Anschlüsse

Nachdem wir die interne Architektur der AVR-Mikrocontroller kennen, können wir uns jetzt mit der externen Beschaltung der wichtigsten Anschlüsse befassen.

Die Versorgungsspannung wird über die Pins Vcc und GND zugeführt und sollte für einen sicheren Betrieb genügend stabilisiert sein. Des Weiteren sollte ein Kondensator (~ 100 nF) zum Abblocken von Hochfrequenz-Störungen, wie sie jedes elektronische Gerät erzeugt, zwischen Vcc und GND vorgesehen werden. Dieser Kondensator ist so nah wie möglich am Mikrocontroller zu positionieren, damit über die Verbindungsleitungen zwischen Kondensator und IC keine weiteren HF-Störungen entstehen.

Die Taktversorgung erfolgt über den internen Oszillator, der über einen Quarz oder einen Keramikresonator auf die gewünschte Taktfrequenz stabilisiert wird (Abbildung 3). Dieser wird mit den Anschlüssen XTAL 1 und XTAL 2 verbunden. Hierbei ist XTAL 1 der Eingang des Oszillators und XTAL 2 der Ausgang. Um einen sicheren Anlauf und stabilen Betrieb zu gewährleisten, sind die Kondensatoren C 1 und C 2 notwendig. Ein Abgriff des erzeugten Taktes ist über eine Treiberstufe in HC-Technologie (High-Speed-CMOS) ohne weiteres möglich. Dabei wird der Taktausgang des Mikrocontrollers (XTAL 2) mit dem Eingang der Treiberstufe verbunden. Der Ausgang des Treibers stellt einen stabilen Takt zur weiteren Verwendung zur Verfügung. Ein direkter Abgriff, d. h. ohne den Puffer, würde den Oszillator zu stark belasten, sodass ein stabiles Schwingverhalten nicht mehr gewährleistet wäre. Aus diesem Grund darf der Ausgang XTAL 2 maximal mit einer HC-Treiberstufe belastet werden.

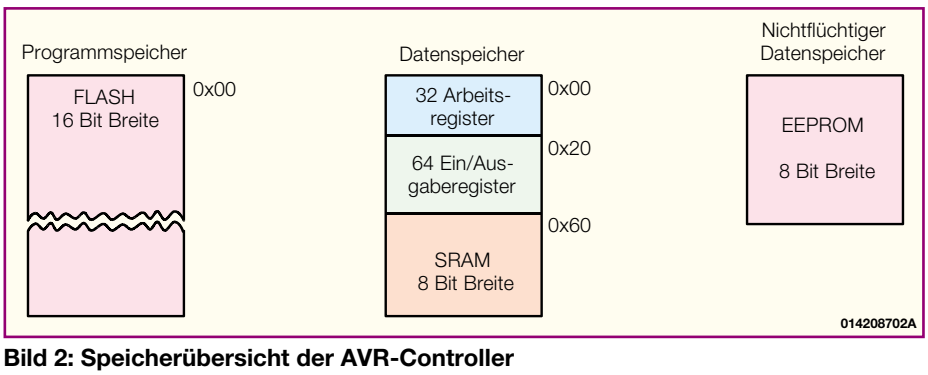

Bild 2: Speicherübersicht der AVR-Controller

Tabelle 1: Funktionsübersicht einiger AVP-Mikrocontroller

	AT90S1200	AT90S2313	AT90S2323/43	AT90S4433	AT90S8515	ATtiny15	ATmega161
Speicher							
Programmspeicher (FLASH)	1 KBytes	2 KBytes	2 KBytes	4 KBytes	8 KBytes	1 KBytes	16 KBytes
Datenspeicher (SRAM)	-	128 Bytes	128 Bytes	128 Bytes	512 Bytes	-	1 KBytes
EEPROM	64 Bytes	128 Bytes	128 Bytes	256 Bytes	512 Bytes	64 Bytes	512 Bytes
Stack	Hardware	SRAM	SRAM	SRAM	SRAM	Hardware	
Peripherie							
Ein-/Ausgänge	15	15	3(AT90S2323) 5 (AT90S2343)	20	32	6	35
8-Bit-Timer/Counter	1	1	1	1	1	2	2
16-Bit-Timer/Counter	-	1	-	1	1	-	1
Watchdog-Timer	Ja	Ja	Ja	Ja	Ja	Ja	Ja
Analog-Komparator	Ja	Ja	-	Ja	Ja	Ja	Ja
Analog-Digital-Umsetzer (10 Bit)	-	-	-	6 Kanäle	-	4 Kanäle	-
UART	-	Ja	-	Ja	Ja	-	Ja
SPI (serielles Interface)	-		-	Ja	Ja	-	Ja
Sonstiges					PWM-Output Brown-Out-Reset	PWM-Output	
Technische Daten							
Max. Taktfrequenz	12 MHz	10 MHz	10 MHz	8 MHz	8 MHz	Intern	8 MHz
Max. Spannungsbereich	2,7 .. 6,0 V	2,7 .. 6,0 V	4,0 .. 6,0 V	4,0 .. 6,0 V	4,0 – 6,0 V	4,0 .. 5,5 V	4,0 .. 5,5 V
Stromaufnahme @ 4 MHz, 3 V, 25 °C							
Active	2,0 mA	2,8 mA	2,4 mA	3,4 mA	3,0 mA	Nicht angegeben	Nicht angegeben
Idle-Mode	0,4 mA	0,8 mA	0,4 mA	1,4 mA	1,0 mA	Nicht angegeben	Nicht angegeben
Power-Down-Mode	< 1 µA	< 1 µA	< 1 µA	< 1 µA	< 1 µA	Nicht angegeben	Nicht angegeben
Gehäuse	PDIP-20, SOIC-20	PDIP-20, SOIC-20	PDIP-8, SOIC-8	PDIP-28, TQFP-32	PDIP-40, PLCC-44, TQFP-44	PDIP-8, SOIC-8	PDIP-40, PLCC-44, TQFP-44

Eine weitere Möglichkeit den Controller zu takten, besteht durch die Einspeisung eines externen Taktsignals über den Eingang XTAL 1 (Abbildung 4). In dieser Betriebsart muss XTAL 2 unbeschaltet bleiben. Diese Möglichkeit findet Verwendung, wenn in einer Schaltung schon ein passendes Taktsignal vorhanden ist und somit weitere externe Komponenten eingespart werden können. Diese Einsparung ist auch bei einigen Typen der AVR-Familie in anderer Weise möglich. Hier kann ein interner RC-Oszillator aktiviert werden, der den Takt ohne externe Bauelemente erzeugt (AT90S1200, ATtiny10/11/12/22/22L).

Beim Zuschalten der Versorgungsspannung ist ein zuverlässiger Reset unbedingt erforderlich, d. h. alle Register müssen auf ihre im Datenblatt angegebenen Startwerte und der Befehlszähler auf den Programmanfang gesetzt werden. Ein solcher Reset wird ausgelöst, wenn der RESET-Anschluss für eine definierte Zeit auf Low-Pegel gehalten wird. Die entsprechende Schaltung ist in Abbildung 5 zu sehen. Das RC-Glied hält das Potential am Reset-Eingang für eine gewisse Zeit auf Low, sodass der Controller seine definierten Startbedingungen einnehmen kann. Im normalen Betrieb liegt dann ein High-Pegel am Eingang an, da der Kondensator sich über den Widerstand aufgeladen hat. Die Diode parallel zum Widerstand gewährleistet ein schnelles Entladen des Elkos im Falle der Abtrennung der Betriebsspannung, damit beim Einschalten immer ein Low-Pegel am Reset-Eingang anliegt. Ein nicht ordnungsgemäßer Reset würde zu unerwünschten Fehlfunktionen des im Mikrocontroller laufenden Programms führen.

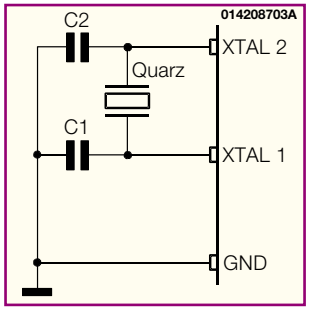

Bild 3: Beschaltung des Oszillators

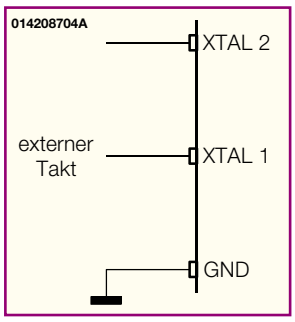

Bild 4: Einspeisung eines externen Taktsignals

Portbeschreibung

Die verschiedenen Typen der AVR-Mikrocontroller verfügen über eine unterschiedliche Anzahl von Ein-/Ausgabeleitungen, über die die angeschlossene Hardware gesteuert werden kann. Außerdem verfügen einige Anschlüsse, je nachdem welche Peripheriekomponenten in den Controller integriert sind, über Alternativfunktionen, wie z. B. PWM-Ausgang, Timersteuerung, Eingang des Analog-Digital-Umsetzers usw.

Diese Alternativfunktionen können den jeweiligen Datenblättern entnommen werden. Im Folgenden wird nur der Betrieb als digitaler Ein-/Ausgabeport betrachtet. Jeder Portpin ist einzeln als Ein- oder Ausgang schaltbar. Ist ein Pin als Eingang definiert, kann zusätzlich ein PullUp-Widerstand aktiviert werden, der im unbeschalteten Zustand einen definierten Pegel erzeugt. Wird der Pin außerhalb des Mikrocontrollers auf Massepotential gelegt, fließt über den PullUp-Widerstand ein nicht zu vernachlässigender Strom. Ein hochohmiger Zustand des Eingangs kann durch das Abschalten des PullUp-Widerstands erreicht werden. Ports mit dieser Eigenschaft werden auch als Tristate-Ports bezeichnet. Die internen PullUp-Widerstände liegen im Bereich von 35 .. 120 kΩ. Ein als Ausgang betriebener Port kann bis zu 20 mA treiben, womit z. B. schon eine Leuchtdiode direkt ansteuerbar ist.

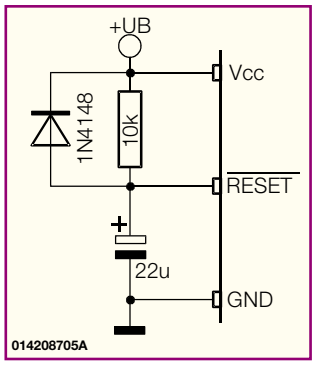

Bild 5: Reset-Schaltung

Hiermit ist die Beschreibung der Hardwareeigenschaften der AVR-Mikrocontroller-Familie bereits abgeschlossen. Im nächsten Teil der Artikelserie wird der Befehlssatz dieser Controller detailliert erläutert. ELV

Internet
AVR-Seite des Herstellers:
www.atmel.com/atmel/products/prod23.htm