Was ist ein Mikrocontroller?

Proseminar Mikrocontroller und eingebettete Systeme WS2014/2015

Bernhard Metz
Lehrstuhl für Echtzeitsysteme und Robotik
Fakultät für Informatik
Technische Universität München
Email: metzb@in.tum.de

Zusammenfassung

Mikrocontroller sind unbesungene Helden des Alltags, der mehr und mehr von den verschiedensten elektronischen Geräte durchdrungen wird, von denen viele ohne Mikrocontroller kaum denkbar wären und doch ist das tatsächliche Konzept eines solchen Gerätes allgemein eher unbekannt. Im Folgenden will ich die Grundzüge, den Aufbau und die vielfältigen Einsatzgebiete von Mikrocontrollern aufzeigen und genauer erläutern.

Index Terms

Mikrocontroller, Mikroprozessor, SoC, System-On-a-Chip, Eingebettete Systeme, Embedded Systems

I. Was ist eigentlich ein Mikrocontroller? (Einführung)

A. Definition

Ein Mikrocontroller ist ein spezieller Mikrorechner, der neben seinem Prozessor diverse andere wichtige Komponenten und Peripherie-elemente auf einem Chip vereint, also ein Ein-Chip-System, jedoch nicht zu verwechseln mit einem System-on-a-chip (SoC, siehe auch unten). Sie werden meist für ihre jeweiligen speziellen Anwendungsfälle angepasst ausgewÃdhlt und sollen im Optimalfall ihre Aufgaben mit möglichst wenig notwendigen externen Bausteinen erfüllen und können in großen Stückzahlen hergestellt werden. Oft vollführen sie Steuerungs- und Kommunikationsaufgaben verschiedenster Art.

Abbildung 1. Renesas 32-bit Mikrocontroller [1]

B. Bedeutung/Motivation

Wie eingangs erwähnt, gewinnen elektronische Geräte immer mehr Bedeutung im Alltag und immer mehr Vorgänge des täglichen Lebens sind zunehmend ohne ihre Hilfestellung kaum oder nur sehr umständlich vorstellbar. Alle diese Geräte sollen naturgemäß möglichst günstig hergestellt werden und trotzdem zuverlässig ihre jeweiligen Aufgaben erfüllen, wodurch man oft zwangsweise bei Mikrocontrollern landet, durch ihre vielfältigen Varianten kann man für all die verschiedenen Anwendungsgebiete passende Geräte erhalten, die die ihnen zustehenden Aufgaben erfüllen, aber auf Grund ihrer passenden Ausstattung in Sachen Rechenleistung, Speicher, Anschlüsse und Ähnliches sehr kostengünstig sind und die Herstellung damit nicht unnötig verteuern.

Seit dem ersten 4-bit Mikrocontroller 1971 (dem Intel 4004) hat sich die Verwendung von Mikrocontrollern stetig erweitert und differenziert, obwohl heute auch 8, 16 und auch erste 32-bit Varianten existieren, werden noch heute, inzwischen noch kostengünstiger als damals, 4-bit Mikrocontroller zum Beispiel für Uhren, verwendet, die für diese Aufgabe noch immer vollkommen ausreichend und bis heute sehr zuverlässig sind.

Um ein Größenverhältnis zu geben, sei der McClean Report von 2014 nach Ismini Scouras in einem Artikel für die EETimes zitiert: „Die Verkaufszahlen von Mikrocontrollern werden im Jahr 2014 um 6% auf 16,1 Mrd. $ -einem Rekordhoch- wachsen, gefolgt von einem 7% bzw. 9% Wachstum in 2015 & 2016. Die Anzahl an verkauften Einheiten wird wahrscheinlich ebenfalls deutlich zunehmen und zwar um 12% auf 18,1 Mrd. Einheiten in diesem Jahr.“ [2]

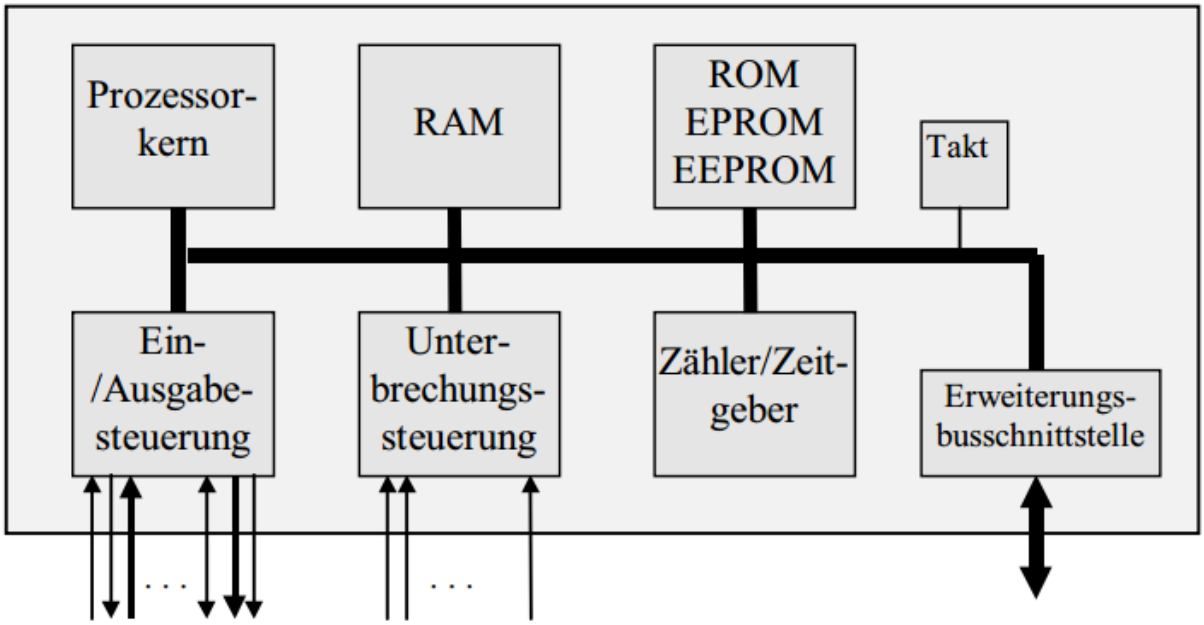

Abbildung 2. Schematischer Aufbau eines Mikrocontroller [3, S.74]

Ein Mikrocontroller besteht vor allem aus einem Prozessor, der die notwendigen Berechnung und andere Operationen durchführt, meist zwei Speicherarten (flüchtiger Schreib-/Lesespeicher und nicht-flüchtiger Festwertspeicher, zweiter ist nicht zwingend notwendig) und einer Ein-/Ausgabesteuerung (E/A). Zusätzlich wird im Normalfall eine Unterbrechungs-(Interrupt-)steuerung benötigt, oft sind auch Zähler und/oder Zeitgeber und diverse Erweiterungsschnittstellen vorhanden, wobei gerade letztere meist aus Kostengründen bewusst knapp gehalten werden.

A. Abgrenzung zum Mikroprozessor

Um Missverständnisse oder Verwechslungen zu vermeiden, soll vor der genauen Erläuterung des Aufbaus hier eine kurze Abgrenzung stattfinden. Der Übergang ist in manchen Fällen durchaus ein wenig fließend, doch im Allgemeinen ist ein Mikroprozessor, wie der Name schon andeutet eben ein Prozessor, der zusätzlich auf dem Chip maximal noch seine Register besitzt, er konzentriert sich vor allem auf die performante Ausführung seiner Rechenoperationen. Ein Mikrocontroller hingegen besitzt ein Prozessor und zusätzlich diverse Elemente auf einem Chip, er kann mit seinen drei Kernelementen (Prozessor, Speicher und Ein-/Ausgabe) bereits als kompletter Mikrorechner eingesetzt werden. Bei ihm geht es meist weniger um tatsächliche Rechenleistung als darum, seine Aufgaben kostengünstig und mit möglichst wenig externen Bauteilen zu erfüllen.

B. Abgrenzung System-on-a-Chip

Der Begriff System-on-a-Chip (SoC) bzw. Ein-Chip-System lässt sich nicht komplett von dem des Mikrocontrollers abgrenzen, da er eine besondere Art Mikrocontroller beschreibt. Dieser muss wie oben erwähnt eigentlich nur die grundlegendsten Bauteile auf seinem Chip besitzen (CPU, Speicher, E/A) und kann zusätzlich externe Komponenten für seine Aufgabe in Anspruch nehmen.

Ein SoC beschreibt den Fall, dass tatsächlich sämtliche benötigten Komponenten eines Systems auf einem einzelnen Chip integriert sind. Dies ist durch die heutige Herstellungstechnik meist relativ einfach möglich, aber in vielen Anwendungsfällen des Mikrocontrollers nicht wünschenswert, weil z.B. auswertende Sensoren im jeweiligen System bereits vorhanden sind und es keine Vorteile bringt, diese stattdessen auf dem Mikrocontroller zu verbauen. Ein SoC kann meist auch als (gut ausgestatteter) Mikrocontroller bezeichnet werden, ein Mikrocontroller aber nicht zwingend als SoC.

C. Der Prozessor

Der Prozessor (CPU, Central Processing Unit) ist das Herzstück eines Mikrocontrollers und ist für Berechnungen und Operationen zuständig. Wie auch andere Prozessoren besteht er aus einer Arithmetik-Logik-Einheit (auch ALU genannt, Arithmetic Logical Unit) die das Rechenwerk des Prozessors darstellt und wie der Name andeutet arithmetischen Berechnungen (also Addition, Subtraktion, Division und Produkt) und logische Operationen (Boolesche Algebra) durchführt und einem Steuerwerk, das den Ablauf der Befehlssteuerung für die korrekte Durchführung der jeweiligen Aufgabe verwaltet.

Hier gibt es verschiedene Architekturen und zwei bestimmende Design-Philosophien, die Auswahl ist sowohl für die Rechenleistung als auch für den Preis entscheidend, ebenso im Falle der eigenen Programmierung des Mikrocontrollers (siehe dazu auch Speicher).

Die zwei konkurrierenden Design-Philosophien sind CISC und RISC. CISC steht für „Complex Instruction Set Computer“ und beschreibt Prozessoren, die relativ viele verschiedene Befehle beherrschen. Aus ihnen entstanden später RISC Prozessoren, da die vielen Befehle von CISC zwar in manchen Fällen das Programmieren vereinfachte, in vielen Fällen aber nicht nötig war und man ein Prozessor mit wenigen Befehlen wesentlich einfacher und schneller sein könnte. Also wurde CISC “entschlankt“ und RISC entstand, dessen Name „Reduced Instruction Set Computer“, mit dem ersten Wort “Reduced“ bereits verdeutlicht was dieses Design abgrenzt. RISC beschreibt also CPUs mit wenigen Befehlen, die dafür einfacher herzustellen und allgemein performanter sind, dafür muss beim Programmieren manchmal mehr Arbeit geleistet werden, da man in CISC normal vorhandene Befehle in RISC mit mehreren umsetzen muss. Bei der Auswahl muss hierbei vor allem auf die voraussichtlichen Befehle geachtet werden, die der Mikrocontroller ausführen wird, sollte es abzusehen sein, dass tatsächlich oft in CISC umgesetzte Befehle genutzt werden, sollte es auch genutzt werden, in anderen Fällen wird wahrscheinlich eher RISC genutzt werden. [4]

Eine weitere wichtige Kennzahl des Prozessors ist die Datenbreite des Prozessorkerns, allgemein gesagt die Architektur des Mikrocontrollers. Hier wird unterschieden zwischen 4-Bit, 8-Bit, 16-Bit und 32-Bit. Es handelt sich um die Datenbreite der internen Datenpfade und ist damit entscheidend für die Leistungsangabe des Prozessors, daher wird diese Kennzahl meist auch genutzt um Mikrocontroller in Leistungsklassen einzuteilen. Ebenso wird sie oft genutzt um Mikrocontrollerfamilien zu bestimmen. Eine Familie wird durch ihren Prozessorkern definiert und die einzelnen Mitglieder besitzen verschiedene Speichergrößen, Peripherieeinheiten oder Erweiterungsbusse um für verschiedene Aufgaben jeweils optimal geeignet zu sein.

Im Gegensatz zu den Prozessoren in normalen PCs ist es allerdings bei Mikrocontrollern nur langsam so, dass die „alten“ schwachen Prozessoren mit geringer Bit-Zahl aussterben. Noch heute gibt es 4-bit Mikrocontroller, doch diese bilden tatsächlich seltene Ausnahmefälle. 8-bit Prozessoren verlieren inzwischen zwar deutlich an Marktanteilen, sind aber noch heute für viele Anwendungszwecke die erste Wahl. Am stärksten wachsend ist inzwischen der 32-bit Markt, da der Preisunterschied zwischen 16 und 32-bit oft geringer ist, als der Leistungsunterschied. Die Taktzahlen sind fÃijr alle Architekturen im Vergleich zu den gewohnten Größen aus dem PC-Markt eher gering und rangieren meist im zweistelligen MHz-Bereich, mit Ausnahmen im 32-bit Bereich die teilweise auch dreistellige MHz-Zahlen aufweisen.

Hier macht sich oft auch der wichtige Kostenfaktor von Mikrocontrollern bemerkbar, oft werden hier veraltete Versionen eigener Mikroprozessoren benutzt, die jedoch speziell angepasst werden. So sind oft Stromsparmodi interessant, teilweise wird sogar bewusst die Leistung reduziert z.B. die Pipelining-Fähigkeit oder gar die Caches entfernt, dies macht den Prozessor spürbar langsamer, jedoch den zeitlichen Ablauf der Befehle wesentlich vorhersagbarer. Ohne Pipelining ist die benötigte Anzahl von Taktzyklen konstant und ohne Caches ist die Ausführungszeit von Speicher- und Ladebefehlen weniger variabel, weil eine Abwicklung über den Cache (die kaum vorhersagbar wäre) deutlich schneller als eine normale Abwicklung über den Arbeitsspeicher passieren würde. [3, S.75ff, S.139ff]

D. Speicher

Wie oben erwähnt gibt es normalerweise zwei Speicherarten, den immer notwendigen flüchtigen Schreib-/Lesespeicher und den optionalen aber in den meisten Fällen vorhandenen nicht-flüchtigen Festwertspeicher. Die Unterteilung spiegelt sich auch in den ablegenden Daten wieder, sie lassen sich in die zwei Kategorien Programm und Daten unterteilen.

Die Daten werden auf dem flüchtigen (Daten gehen beim Abschalten des Mikrocontrollers verloren) Schreib-/Lesespeicher gehalten, hierbei sind sämtliche Informationen gemeint, die während des Betriebes anfallen, seien es Sensorenwerte, daraus errechnete Einstellungswerte, Prozesszustände oder Änliches. Dieser Speicher ist auch wesentlich schneller als der Festwertpeicher. Es wird prinzipiell zwischen zwei Arten entschieden, ein statischer Schreib-/Lesespeicher nutzt elektronische Schaltungen, sogenannte Flip-Flops, um Informationen zu speichern. Hierzu muss ständig Spannung anliegen, man benötigt mehrere Transistoren pro Flip-Flop, dafür ist der Zugriff sehr schnell. Da die Speichergrößen in Mikrocontrollern sehr klein sind und sich im Bytes bzw. KBytes bereich bewegen, wird diese Methode trotz des hÃűheren Preises oft angewandt.

Ein dynamischer Schreib-/Lesespeicher nutzt Kondensatoren zum Speichern von Informationen. Diese verlieren die Informationen allerdings nach kurzer Zeit durch Selbstentladung oder wenn die Information ausgelesen wird. Die Informationen müssen also sowohl periodisch als auch nach jedem Zugriff neu geschrieben werden, wofür eine aufwendige Steuerlogik erforderlich ist, die den Zugriff verlangsamt. Diese Variante findet vor allem in Arbeitsspeichern in PCs Verwendung, weil die dortigen Speichergrößen (GB-Bereich) durch statische Speicher nicht wirtschaftlich interessant realisierbar wären.

Die Programmdaten hingegen werden auf dem nicht-flüchtigen Festwertspeicher gehalten. Dieser ist meist Teil des Mikrocontrollers, kann aber auch extern angeschlossen werden. Hier lassen sich vier verschiedene Speichervarianten unterscheiden, die auch alle in den passenden Anwendungen noch heute Gebrauch finden.

Die ersten Mikrocontroller (aber viele auch noch heute) wurden mit einem ROM (Read Only Memory) Speicher hergestellt. Dieser Speicher wird permanent bereits bei der Herstellung fest auf den Chip mithilfe einer Herstellungsmaske (daher auch “Maskenprogrammiert“ genannt) programmiert und kann im Nachhinein nicht mehr verändert werden. Dafür ist der Speicher einfach, kompakt und preisgünstig zu realisieren und besitzt hohe Geschwindigkeit und Zuverlässigkeit (z.B. gegenüber extremen Temperaturen). Auf Grund der Kosten der Herstellungsmaske ist diese Variante aber vor allem für Großserien (mind. 20.000) interessant. Außerdem muss das Programm “aus dem Haus“ gegeben werden und kann auch nicht im Systemumfeld getestet werden.

Für kleinere SeriengrÃűÃ§en sind diese Kosten jedoch sehr problematisch, da der Preis pro Mikrocontroller dadurch schnell zunimmt, daher gibt es für diese als Alternative PROM-Speicher (Programmable Read Only Memory). Diese sind wie die vorherige Variante ausschließlich lesbar, doch muss das Programm nicht schon bei der Herstellung über eine Maske programmiert werden, sondern kann im Nachhinein im eigenen Haus über ein Programmiergerät beschrieben werden. Einmal programmiert ist aber auch diese Variante permanent! Es entfallen die Kosten für eine Herstellungsmaske, dafür

ist der Speicher selber teurer und es wird ein Programmiergerät benötigt, das jedoch wesentlich preisgünstiger als die Herstellung entsprechender Programmierungsmasken ist. Es muss unter UmstÃďnden hier also auch in Betracht gezogen werden, ob man nur einmal PROM nutzt oder doch mehrere Male.

Für Kleinstserien oder spezielle Projekte bei denen die Löschbarkeit des Programmes von hoher Bedeutung ist gibt es OTROM/OTP (One Time PROM/One Time Programmable) bzw. EPROM (Erasable PROM), beide Varianten beschreiben denselben Speicher, der ähnlich eines PROM-Speichers per Programmiergerät beschrieben werden kann, das Programm aber in speziellen Transistoren speichert, die durch Bestrahlung von UV-Licht gelöscht werden können. EPROM-Speicher sind langsamer, können aber sehr nützlich sein, wenn eine Neu-Beschreibung bzw. eine Änderung des Programms für die zu erledigende Aufgabe von hoher Bedeutung sind. OTROM-Mikrocontroller nutzen denselben Speicher, erhalten jedoch kein aufwendiges Keramikgehäuse mit dem zum Löschen notwendigen Quarzglasfenster und sind dadurch nur einmalig beschreibbar. Dies ist nur selten für Prototypen oder manchmal auch für Einzelherstellungen interessant.

Die neueste Entwicklung sind jedoch elektronisch löschbare Speicher, FlashRAM und EEPROM (Electrically Erasable PROM) die einfach durch eine Löschspannung zu löschen sind und daher kein spezielles Gehäuse benötigen. Sie sind damit sehr flexibel und bei kleineren bis mittleren Stückzahlen heute auch dominierend, für Großserien sind aber auch noch heute vor allem ROM-Varianten gebräuchlich. Der Unterschied zwischen den beiden Varianten liegt in ihrer Löschlogik, FlashRAM lässt sich nur in großen Blöcken oder gar komplett löschen und ist daher für Verwendungszwecke interessant, bei denen der Mikrocontroller häufig neue Programmversionen oder komplett andere Programme erhält, während bei einem EEPROM auch einzelne Zellen gezielt löschbar sind, wodurch er für Aufgaben interessant wird, bei denen diverse einzelne Daten gespeichert und regelmäßig aktualisiert werden müssen, wie z.B. Konfigurationsdaten. [3, S.77, S.182ff]

E. Ein-/Ausgabe

Ohne Verbindung zur Außenwelt können die wenigsten Mikrocontroller die ihnen zugeteilte Aufgabe erledigen. Diese Kommunikation erfolgt über die verschiedenen Ein- und Ausgabekanäle entweder digital oder über verschiedene analoge Signale, meist einer Spannung.

Im Falle digitaler Ein-/Ausgabesignale ist normalerweise kein weiterer Zwischenschritt mehr notwendig, da der Mikrocontroller diese Signale direkt verwenden kann. Für analoge Signale werden Analog/Digital-Wandler (A/D-Wandler) für Eingangssignale benötigt um verarbeitbare digitale Signale zu erhalten. In die Gegenrichtung benötigt man Digital/Analog-Wandler (D/A-Wandler) die die digitalen Informationen, die der Mikrocontroller anderen Komponenten mitteilen will in für sie verständliche analoge Signale umwandeln können.

Weitere Unterscheidungen gibt es im Bezug auf die Adressierung der Ein- und Ausgabeports, diese können entweder nach der sogenannten isolierten Adressierung behandelt werden, bei der die Ein-/Ausgabekanäle einen fest reservierten Speicherbereich besitzen oder nach der sogenannten gemeinsamen Adressierung, bei der sie sich den Speicherbereich teilen. Digitale Ein/Ausgabeeinheiten können außerdem je nach Übertragungsart in serielle und parallele Einheiten unterteilt werden. [3, S.77f, S.142ff]

F. Zeitgeber & Zähler

Diese Komponenten ermöglichen es dem Mikrocontroller Ereignisse zu zählen oder Zeiten zu messen, sie können als Teil des Mikrocontrollers implementiert, aber auch als externe Komponente angeschlossen werden.

Zeitgeber sind oft mit Zählern verbunden, da das Messen von Zeit in Mikrocontroller meist über das Zählen eines regelmäßigen Taktes funktioniert, je nach Prozessorart kann z.B. einfach der Prozessortakt gezählt werden und in Zeit umgerechnet werden. Zeitgeber und Zähler sind für viele

Anwendungen notwendig und haben oft verschiedene Funktionen, so können sie genutzt werden um das System nach gegebener Zeit über ein Interrupt-Signal zu “wecken“. Vor allem für Echtzeitanwendungen werden sie besonders benötigt.

Zähler besitzen die Möglichkeit auf oder ab zu zählen. Die CPU kann hierbei den Startwert festlegen, der auch bei einer Neuinitialisierung (wenn null erreicht wird) wieder neu geladen wird. So kann durch das stetige Herabzählen vom selben Wert zum Beispiel auch ein regelmäßiger Impuls erzeugt werden. Zusätzlich kann die CPU auch über das Zählerstandsregister jederzeit den aktuellen Zählerstand auslesen. Gezählt werden hierbei entweder Ereignisse, externe Signale über den Zählereingang oder Flanken (Signalübergänge). [3, S.75ff, S.139ff]

G. Interrupt-/Unterbrechungssteuerung

Der Begriff Unterbrechung (im Weiteren wird der auch im Deutschen gebräuchliche Begriff Interrupt verwendet) mag erst einmal etwas negativ beziehungsweise nach einem Fehler klingen(was auch eine Ursache für ein Interrupt-Signal sein kann), ist aber ein wichtiger Bestandteil eines Mikrocontrollers beziehungsweise eines jeden Computersystemes.

Interrupts können von verschiedenen Komponenten benutzt werden um das laufende Programm zu unterbrechen, die CPU speichert sämtliche notwendigen Werte zum späteren Fortführen des Programms zwischen und ruft die passende Interrupt Service Routine auf, führt diese durch und macht danach mit der Ausführung des Programmes an der unterbrochenen Stelle weiter. Es ist sozusagen eine Möglichkeit zum Aufruf eines Unterprogramms. Interne Komponenten nutzen Interrupts um der CPU geänderte Zustände zu signalisieren, zum Beispiel dass Daten zum Bearbeiten vorhanden sind, über Interrupt-Eingänge können auch externe Komponenten ein Interrupt-Signal senden, wenn sie Daten aus dem Mikrocontroller benötigen oder Änliches, das laufende Programm kann sie nutzen um auf spezielle Funktionen zuzugreifen und manchmal treten sie auch tatsächlich durch Fehler, wie zum Beispiel eine versuchte Division durch Null, auf. [3, S.79, S.168ff]

H. Andere

Für die meisten Mikrocontroller von großer Bedeutung ist ein sogenannter Erweiterungsbus der genutzt werden kann um externe Komponenten anzuschließen. Um der begrenzten Anzahl gerecht zu werden, wird oft Daten- oder Daten-/Adress-Multiplexing genutzt, um die gleiche Anzahl an Daten über weniger Leitungen transportieren zu können.

Je nach Verwendungszweck kann eine DMA (Direct Memory Access) Komponente von Interesse sein, die zum Transport von Daten spezialisiert ist. Bis auf die Intialisierung durch die CPU benötigt sie keine weiteren Befehle des Prozessors um Daten zu lesen und zu schreiben und kann damit besonders bei schwachen Prozessoren oder häufig auftretenden Lese-/Schreiboperationen die CPU deutlich entlasten.

Für spezielle Anwendungen ist unter UmstÃnden ein sogenannter Watchdog interessant, der den korrekten Ablauf eines Mikrocontroller-Programms überprüft. Dies tut er im Normalfall in dem er regelmäßig den Wert eines Zeitgebers überprüft, denn die CPU im Laufe ihres Programms regelmäßig zurücksetzen soll. Ist dieser Zeitgeber einmal nicht zurückgesetzt, nutzt der Watchdog einen Interrupt um den Mikrocontroller neu zu starten. [3, S.179f, S.197ff, 201ff]

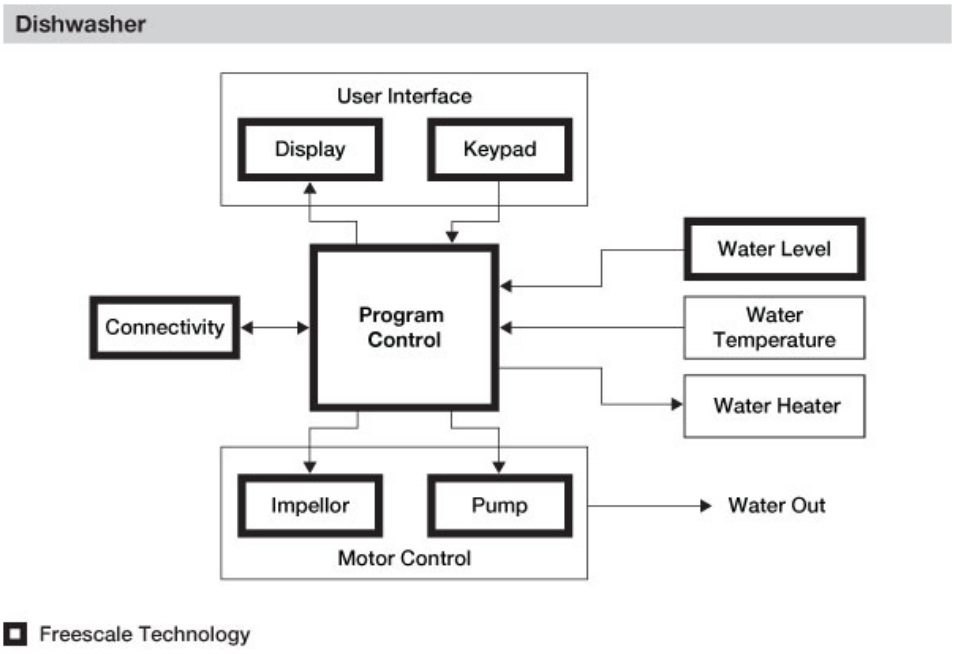

Abbildung 3. Anwendungsdiagramm eines Freescale Mikrocontrollers in einer Geschirrspülmaschine [5]

Mikrocontroller finden heutzutage in sehr vielen technischen Gebrauchsartikeln unseres Alltags in irgendeiner Form eine Anwendung und die Einsatzbereiche sind dadurch maximal breit gefächert, weil sie in Leistung und Ausstattung leicht dem jeweiligen Einsatz anzupassen sind und ihre Aufgabe in Sachen Kosten und Stromverbrauch sehr sparsam erledigen. Eine Liste sämtlicher technischer Geräte die Mikrocontroller verwenden dürfte kaum umfassend aufstellbar sein, sie finden sich im Haushalt, Kraftfahrzeugen, Herstellungsmaschinen, Büro- und Unterhaltungselektronik bis hin zu Handys und Uhren wieder.

A. Messen, Stellen, Regeln

Exemplarisch sei ein Anwendungsfeld etwas genauer beschrieben, dessen Grundzüge sich wahrscheinlich auf viele Mikrocontroller-Aufgaben übertragen lassen. Oft geht es darum diverse Prozesse zu steuern, teilweise automatisch oder auch nach Benutzereingaben. Hierbei müssen Mikrocontroller meist über ihre Ein-/Ausgabekänale oder auch Erweiterungsanschlüsse mit diversen Sensoren z.B. für Temperatur zusammenarbeiten. Von diesem erhält die CPU Eingangsdaten, die dann nach verschiedenen Vorgaben in nutzbare elektrische Werte umgerechnet werden müssen.

Anhand dieser Werte und eventueller Vergleichswerte erfolgt dann ein Stellenvorgang. Es kann zum Beispiel ein Drucksensorwert eingelesen werden, mit einem gespeicherten Maximalwert verglichen werden und im Falle, dass dieser erreicht sein sollte ein Steuersignal an ein ebenfalls an den Mikrocontroller angeschlossenes Druckentlassungsventil gesendet werden. Ein alternatives Beispiel aus dem Haushalt wäre auch eine Heizungsanlage, die ständig über einen Temperatursensor die aktuelle Temperatur misst, sie mit dem vom Benutzer eingestellten Wert vergleicht und je nach Ergebnis entsprechende Signale an die Steuerung der Heizanlage schickt um die Temperatur zu erhöhen / zu senken.

Der Regelvorgang ist danach mehr ein Korrekturvorgang des Stellens, bei dem über Sensoren der Istwert mit dem Sollwert verglichen wird um eventuelle Korrekturen vorzunehmen, die nötig sind, weil v.a. durch die Diskrepanz zwischen elektrischen Eingangssignalen und den berechneten physikalischen Ausgabegrößen oder auch einfach durch den Einfluss von Störgrößen Fehler entstehen können. [3, S.82ff]

IV. Beispiele

Hier sollen kurz Beispiele aus der Welt der Mikrocontroller vorgestellt werden um das im obigen Text dargestellte Bild anhand realer Beispiele zu verdeutlichen. Die Anzahl verschiedener Mikrocontroller ist gemäß den vielen verschiedenen Anwendungszweck sehr groß und daher eine repräsentative Darstellung aller Mikrocontroller kaum möglich.

A. Arduino (Uno) bzw. ATMega328

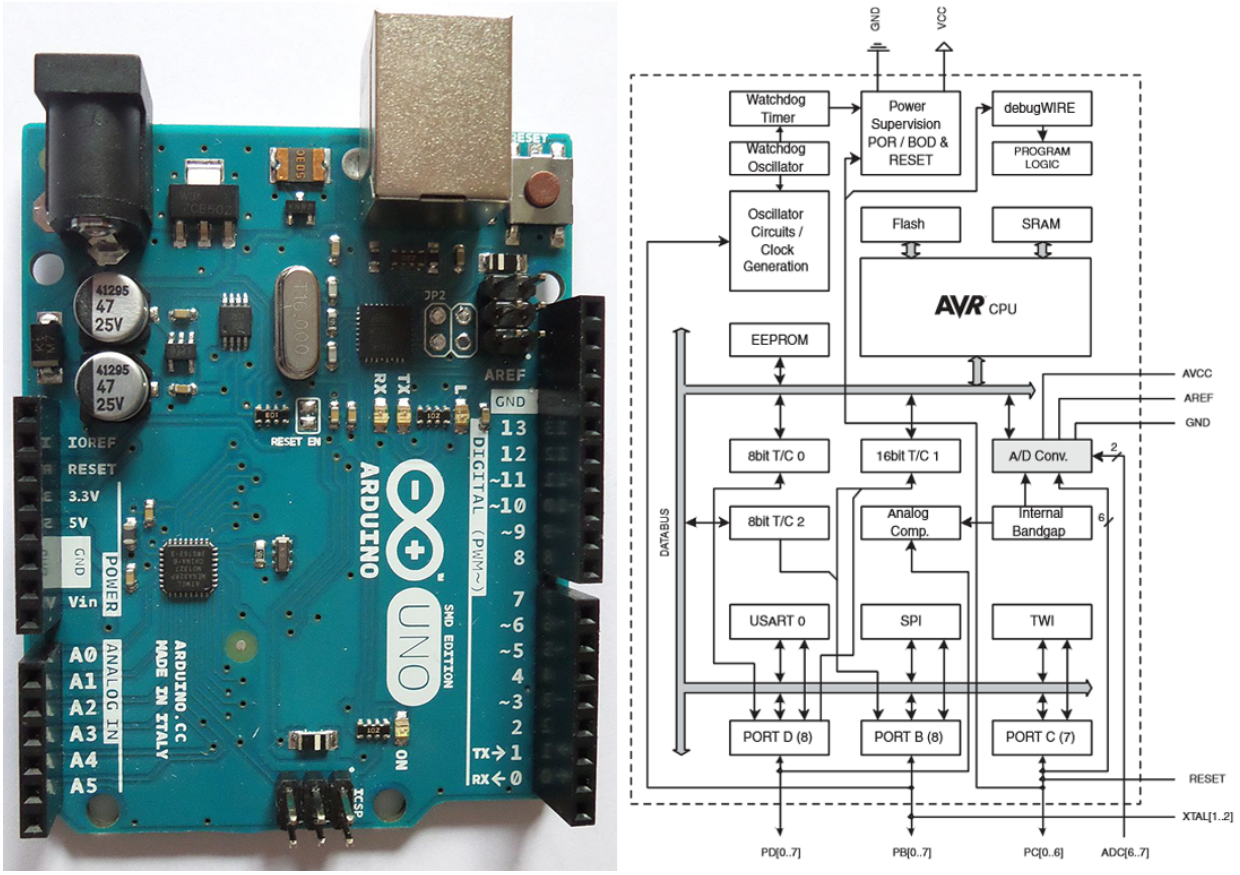

Abbildung 4. Arduino R3 und Block-Diagramm des ATmega328 [6] [7]

Das 2006 gestartete Arduino Projekt dürfte einer der wichtigsten Schritte sein, Mikrocontroller aus der Welt der Elektrotechniker, Informatiker usw. in die Haushalte von technisch interessierten (aber nicht zwingend technisch studierten) Privatleuten zu bringen. Der Begriff Arduino beschreibt verschiedene Versionen des Arduino Boards mit verschiedenen eingesetzt Mikrocontrollern, doch vereint sie alle die ebenfalls zum Arduino Projekt gehörende Entwicklungsumgebung (IDE), mit

der man ohne große Vorkenntnisse erste Programme (im Arduino-Bereich sogenannte „Sketche“) schreiben kann, die dann von der IDE selbst in für den Mikrocontroller sogenannten “Steuercode“ umgewandelt werden. Wichtig ist hierbei zu beachten, dass der Arduino im Gegensatz zum Raspberry Pi oder ähnlichen Einplatinencomputer nicht einen Mini-PC für verschiedene Zwecke ersetzen soll, sondern tatsächlich einfach in die Welt der Mikrocontroller einführen soll, indem man ihn ähnlich einem Mikrocontroller nutzt, nämlich über das Schreiben eines Programmcodes, das verschiedene Ein-/Ausgabesignale über die Ports des Arduino für seinen Ablauf nutzt. Entscheidend für die einfache Benutzung ist hierbei der für Mikrocontroller sehr ungewöhnliche USB-Anschluss, über den man die am Computer verfassten Programme als Programm auf den Arduino-Mikrocontroller übertragen kann. Die Fülle an möglichen Projekten und dazu verfügbaren Tutorials/Anleitungen oder auch Begleitliteratur ist riesig.

Hier soll der Arduino Uno R3 näher betrachtet werden, der sozusagen das Standard-Board des Arduino ist, es wird am häufigsten verkauft und viele den Arduino behandelnde Bücher nutzen diese Version als Beispiel für ihre Ausführungen. Dieser nutzt den ATMega328 von Atmel einen 8-bit Mikrocontroller mit einem Atmel AVR Prozessor mit einer maximalen Taktrate von 20 MHz. Er besitzt 32KB nicht-flüchtigen Flash Speicher, 1 KB ebenfalls nicht-flüchtigen EEPROM Speicher und 2KB flüchtigen SRAM (Static Random-Access-Memory). Im Falle des Arduino wird der EEPROM für grundlegende Systemeinstellungen und ähnliche normalerweise auf lange Zeit genutzte Daten benutzt während der größere Flash-Speicher für vom Nutzer in der IDE verfasste Arduino-Programme genutzt wird. Der ATmega328 besitzt 23 Ein-/Ausgabe Ports, im Arduino Uno werden 14 digitale Ein-/Ausgabeports sowie 6 analogie Eingänge zur Verfügung gestellt. Zusätzlich besitzt er noch I^2C, SPI und ICSP Anschlüsse. [8] [7]

B. MC68332

Als zweites Beispiel sei ein „normaler“ Mikrocontroller vorgestellt, der MC68332 ein 32-bit Mikrocontroller von Freescale (ehemals Motorola, 2004 ausgegliedert) erwähnt. Als Prozessorkern wird eine CPU32 Variante der Freescale 680XX Mikroprozessor-Reihe mit einer Takfrequenz von bis zu 25 MHz verwendet. Spezielle Features des MC68332 sind zum Beispiel das in vielen Freescale 16/32-bit Mikrocontroller verwendete System Integration Module, das u.a. ein Zeitsignal und verschiedene Schutzmechanismen für das System zur Verfügung stellt und außerdem als Erweiterungsbus dient.

Außerdem besitzt der MC68332 eine eigene Time Processing Unit, einen zum Teil selbständigen Koprozessor, der praktisch die Zähler-/Zeitgeberkomponente ersetzt, er besitzt einen eigenen ROM-Speicher mit verschiedenen vorgebenen Funktionen, mehrere eigene Timer, einen Task Scheduler, einen eigenen RAM-Speicher für die verschiedenen Zeit-/Zählwerte und exklusive Anschlüsse. Dieser Koprozessor kann verschiedene Aufgaben selbständig ohne die Mikrcokontroller-CPU erledigen und ist daher eine sehr mächtige Zähler-/Zeitgebereinheit, die den MC68332 für verschiedene Aufgaben zum Beispiel Steuerungsaufgaben speziell geeignet macht.

Der MC68332 besitzt 2 KB statischen RAM Speicher und ist ein Beispiel eines Mikrocontrollers der keinen eigenständigen Festwertspeicher besitzt. Die Ein-/Ausgabe umfasst vier parallele Einheiten mit ingesamt 31 Bit. Zusätzlich gibt es noch das QSM, das zwei serielle Interfaces bietet, ein SCI (Serial Communication Interface) zur asynchronen Übertragung und ein QSPI zur synchronen Übertragung (von SPI, Serial Peripheral Interface, einem von Motorola entwickelten Standard für synchrone serielle Datenübertragung). [3, S.225ff] [9]

V. Fazit / Zukunftsaussichten

Mikrocontroller sind unbesungene Helden des Alltags. Ihre Bedeutung für unsere zunehmend von technischen Geräten aller Art beherrschten Gesellschaft, die all diese Geräte auch noch möglichst billig haben will und viele davon ohnehin sehr schnell wieder wegwirft kann kaum hoch genug eingeschätzt werden. Sie erfüllen unvorstellbar viele Aufgaben leise und heimlich im Stillen, ohne große Kosten zu verursachen, weder bei der Herstellung noch beim Stromverbrauch und passen damit perfekt in unsere Wegwerfgesellschaft mit stetig wachsenden Strompreisen. Auf Grund der begrenzten Anforderungen an ihre Rechenleistung sind sie lange haltbar, Mikrocontroller verrichten ihren Dienst oft über viele Jahre lang und das in den meisten Fällen zuverlässig und ohne Ausfälle, auch durch ihren simpel gehaltenen Aufbau, der ihren Einsatz vereinfacht. Dementsprechend wird ihre Anzahl nur noch weiter wachsen, es wurde in der Einleitung bereits der McClean Report zitiert, der für die nächsten Jahre von einem deutlichen Wachstum am Volumen an Mikrocontrollern ausgeht.

Die Vielfalt der Mikrocontroller wird in Zukunft weiter wachsen und sich dabei vor allem an den sich verändernden Anforderungen orientieren, so gibt es zum Beispiel schon erste Multi-Core Mikrocontroller mit mehreren Prozessorkernen, die verschiedene Aufgaben untereinander aufteilen und für Echtzeitanwendungen interessant sein können. Auch in der Programmierung gibt es Veränderung, sind bisher C (weil sehr Hardwarenahe Programmierung möglich) und im Gegensatz zum PC-Bereich auch noch Assembler vertreten, werden inzwischen teilweise auch Java, speziell für Mikrocontroller entwickelte Programmiersprachen oder auch grafische Entwicklungsumgebungen genutzt.

Literatur

[1] *Renesas Website*, Renesas. [Online]. Available: http://www.renesas.com/press/news/2014/news20140226.jsp

[2] I. Scouras, *Microcontroller Market Resurges*, EETimes, August 2014. [Online]. Available: http://www.eetimes.com/document.asp?doc_id=1323578

[3] T. Ungerer and U. Brinkschulte, *Mikrocontroller und Mikroprozessoren*, 3rd ed. Berlin Heidelberg New York: Springer-Verlag, 2010.

[4] *Mikrocontroller.net*, Mikrocontroller. [Online]. Available: http://www.mikrocontroller.net/articles/CISCbzw./RISC

[5] *Freescale Website*, Freescale. [Online]. Available: http://www.freescale.com/webapp/sps/site/application.jsp?code=APLDWSH

[6] *Arduino Wikipedia-Artikel*, Wikipedia. [Online]. Available: http://de.wikipedia.org/wiki/Arduino-Plattform

[7] *ATmel328 Documentation*, Atmel. [Online]. Available: http://www.atmel.com/Images/doc8161.pdf

[8] E. Bartmann, *Die elektronische Welt mit Arduino entdecken.* Köln: O'Reilly Germany, 2014.

[9] *Freescale MC68332 Documentation*, Freescale. [Online]. Available: http://cache.freescale.com/files/microcontrollers/doc/user_guide/MC68332UM.pdf?fasp=1

Abbildungsverzeichnis