1 Mikrocontroller

Ein Mikrocontroller (µC bzw. uC) hat gegenüber einem uP auch noch so viel Speicher und Peripherie auf demselben Baustein (IC, Integrated Circuit), dass zumindest für Standardaufgaben keine weiteren digitalen Bauelemente mehr benötigt werden. Einen typischen Aufbau zeigt das Blockschaltbild der MSC-48 Serie (siehe Unterlagen) mit den Vertretern 8048 etc. Dies waren nicht die ersten, aber doch frühe uC (ca. 1977).
Diese uC hatten neben dem uP nur noch den Programmspeicher, den Datenspeicher, einen einzigen Zeitgeber (Timer) sowie einige Ein-/Ausgabeleitungen integriert. Heutige uC (2016) haben sehr viel mehr Peripherie mit integriert. Für den im Praktikum verwendeten uC kann dafür das Blockdiagramm (UM, Fig 4) als gutes Beispiel dienen.

1.1 Speicherarchitekturen bei uC

Fast alle heutigen uC fallen entweder in die Kategorie von-Neumann oder *modifiziertes* Harvard. Bei einer reinen Harvard-Architektur kann der Programmspeicher nur zur Speicherung des ausführbaren Programms benutzt werden. In der modifizierten Variante ist es auch möglich, dort Konstanten unterzubringen, d.h. das Programm kann (wenn auch deutlich langsamer als aus dem Datenspeicher) Werte aus dem Programmspeicher lesen. Einige derartige uC erlauben sogar das Löschen und Neubeschreiben von Teilen des Programmspeichers unter Kontrolle des Programms. Dies wird dann als *Self Programming* bezeichnet und ist natürlich nur dann möglich, wenn die Technologie des Speichers das auch prinzipiell erlaubt (siehe Flash, EEPROM).
Auch bei der modifizierten Harvardarchitektur ist es aber nicht möglich, aus dem Datenspeicher ein Programm auszuführen.
Bei den ARM-Rechenkernen wird eine von-Neumann Architektur verwendet.

1.2 Externer Speicher (Programm/Daten)

Viele uC können bei Bedarf mit externem Speicher erweitert werden. In dem Fall werden natürlich einige Anschlüsse für die Anbindung benötigt und gehen damit für andere Aufgaben verloren. Zudem ist der Zugriff auf den externen Speicher oft deutlich langsamer als auf den internen Speicher. Prozessoren, die mit der modifizierten Harvard-Architektur arbeiten können oft nur um Datenspeicher erweitert werden.
Alle Mitglieder der im Praktikum verwendeten Serie LPC43xx können ebenfalls um externen Speicher erweitert werden. Allerdings müssen dafür dann auch entsprechende Anschlüsse reserviert werden. Aus diesem Grund können die Mitglieder mit vergleichsweise wenigen

Anschlüssen (100) nur um Speicher erweitert werden, der seriell angesteuert wird und damit nur sehr wenige Leitungen benötigt (4). Das verlängert dann natürlich die Zugriffszeiten. Diese µC haben einen sehr kleinen Cache speziell für den seriellen Speicher eingebaut, damit wenigstens kurze Programmschleifen nicht immer wieder aus diesem Speicher geladen werden müssen.
Der Anschluss eines externen Speichers zur allgemeinen Verwendung (unterschiedslos Programm- und Datenspeicher) ist aufgrund des dann deutlich erhöhten Anschlussbedarfs (bis zu 60 Anschlüsse) nicht bei allen Mitgliedern dieser Serie möglich.

2 Speichertechnologien

Für die einzelnen Speicherbereiche mit ihren unterschiedlichen Aufgaben gibt es verschiedene geeignete Technologien.

2.1 RAM

Speicher, der sehr schnell sein muß und zugleich beliebig oft sowohl beschreibbar als auch lesbar sein muß, heißt RAM (Random Access Memory). Diese Speicher werden in zwei verschiedenen Technologien gefertigt, einmal als DRAM (Dynamic RAM) und einmal als SRAM (Static RAM). DRAM ist wesentlich preisgünstiger zu fertigen, allerdings etwas langsamer und empfindlicher als SRAM. DRAM wird in PCs als Hauptspeicher verwendet. Derzeit (2016) kann ein einziger DRAM-Schaltkreis ca. 4-8 Gbit speichern.
Allerdings ist es aus technologischen Gründen nicht leicht, DRAM auf demselben Chip wie den Rest des µC unterzubringen. Aus diesem Grund wird DRAM so gut wie immer (2016) extern angeschlossen. Dafür existieren verschiedene Standardschnittstellen (DDRx).
SRAM wird in PCs als Cache-Speicher (im µP) sowie in µC mit von-Neumann Architektur auch als Hauptspeicher verwendet.
Typische Größen für einen solchen in einem uC integrierten Speicher betragen zwischen 4 kB und 256 kB.

2.2 ROM

Ein ROM (Read Only Memory) ist ein Speicher, der nur gelesen werden kann. Sein Inhalt wird bereits bei der Herstellung des Schaltkreises festgelegt und ist im Betrieb unveränderlich. ROMs sind der billigste nichtflüchtige Speicher, allerdings lohnen sich ROMs nur bei sehr hohen Stückzahlen (>100.000) und wenn zugleich sicher ist, dass der Inhalt fehlerfrei ist. Ein Beispiel dafür ist der Programmspeicher eines Taschenrechners.

2.3 PROM

Ein PROM (Programmable ROM) ist ein Speicher, der genau einmal beschrieben werden kann und danach unveränderlich ist. Die technologische Realisierung hat dabei mit einem ROM nichts zu tun. Die meisten uC haben keinen derartigen Speicher bzw. er ist nur für den Hersteller, nicht aber den Anwender sichtbar.
Damit werden dann in der Regel grundlegende Eigenschaften des uC einmalig festgelegt. Diese bleiben dann über die gesamte Lebensdauer des Produkts unveränderbar. Übliche Anwendungsgebiete sind Manipulationsschutz, Verschlüsselungen oder die Kennzeichnung jedes einzelnen Produkts mit einer herstellerseitig einmalig vergebene Nummer.

2.4 EEPROM

Das EEPROM (Electrically Erasable PROM) ist ein nichtflüchtiger Speicher, der jedoch unter speziellen Bedingungen (hohe Spannung) auch im eingebauten Zustand wieder gelöscht werden kann. Dabei können einzelne Speicherstellen gezielt gelöscht werden, wobei der Rest unverändert bleibt. Die Löschzeit für eine einzelne Speicherstelle beträgt einige Millisekunden.

Heutige EEPROMs können je nach Ausführung zwischen 100.000 und 1.000.000 Mal gelöscht werden, bevor sie Defekte zeigen. EEPROMs werden gerne für nichtflüchtige Daten eingesetzt, die sich während der Gerätelebensdauer oft ändern oder bei denen immer nur wenige Speicherstellen mit anderen Werten versehen wird. Beispiele dafür sind Kalibrierungsdaten, wenn das Gerät im Betrieb rekalibriert werden kann.

2.5 Flash

Das Flash („Blitz“) ist ein EEPROM, bei dem jedoch immer ein ganzer Speicherteil (als *Sektor* bezeichnet) auf einmal gelöscht wird. Dazu wird aber nicht mehr Zeit benötigt als zum Löschen einer einzigen Speicherstelle, daher der Name „Blitz“. Nachteilig ist nur, dass nicht einzelne Speicherstellen separat gelöscht werden können. Das Flash eignet sich für Daten, die lange Zeit insgesamt unverändert bleiben oder für die Aufnahme großer Datenmengen, da dann ohnehin immer wieder mit dem Löschen eines ganzen Bereichs neuer Platz geschaffen werden muss. Flash ist heute unter den nichtflüchtigen Speichern der Standard. Anwendungsbeispiele: USB-Stick, MP3-Player, BIOS im PC, Programmspeicher in uC.
Flash ist wie DRAM sehr preisgünstig zu fertigen und es kann vor allem problemlos auf dem gleichen Chip wie der Rechenkern und die Peripherie gefertigt werden. Nahezu alle am Markt befindlichen uC haben daher Flash mit auf den Chip.

3 Programmiermethoden

Auch nichtflüchtiger Speicher muss zu irgendeinem Zeitpunkt einmal programmiert werden. Dafür gibt es mehrere Möglichkeiten.

3.1 Programmierung bei der Herstellung

Sowohl das ROM als auch das Flash können bereits bei der Herstellung programmiert werden. Beim ROM ist das sowieso die einzige Möglichkeit. Diese Methode ist bei sehr großen Stückzahlen (10000+) interessant, weil man sich dabei zum einen die reine Programmierzeit und das zugehörige Handling als Kosten spart. Außerdem wird der Inhalt dann schon beim Hersteller geprüft und es entfällt eine Defektquelle.

3.2 Programmierung im System (spezielle Schnittstelle)

Bei uC, die mit Flash als Speicher arbeiten, gibt es fast immer auch die Möglichkeit, diesen Speicher im eingebauten Zustand über eine spezielle Schnittstelle zu programmieren. Dies wird *In-System-Programming* (ISP) genannt (Abbildung 1).

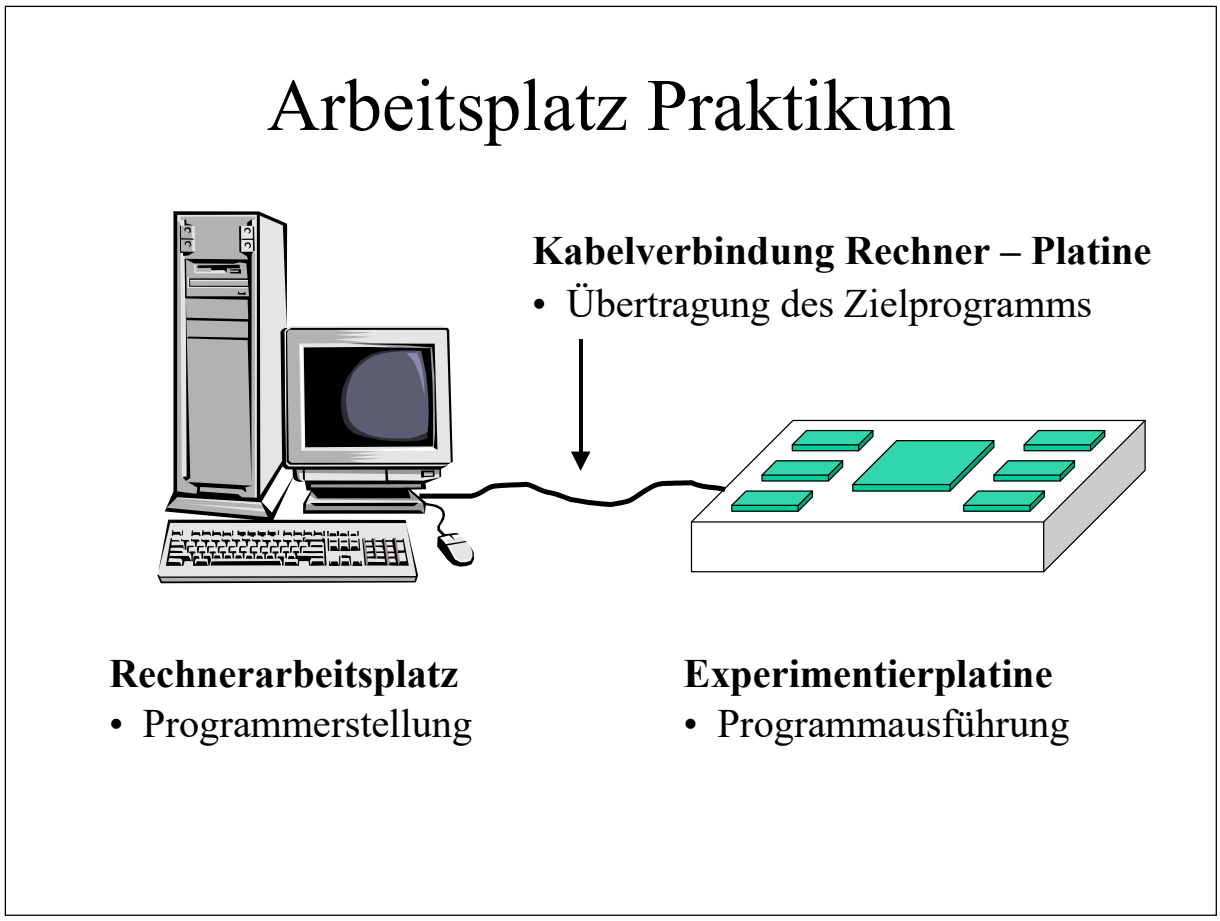

Abbildung 1: Programmierung mit der ISP-Methode (hier am Beispiel der Programmentwicklung)

Die dafür vorgesehene Schnittstelle belegt natürlich Anschlüsse, die möglicherweise ganz oder teilweise für andere Funktionen verlorengehen. Aus diesem Grund sind die meisten dieser Schnittstellen seriell und damit deutlich langsamer als die parallele Programmierung im Programmiergerät. Typischerweise werden 4 Anschlüsse benötigt. Früher (bis ca. 2004) waren die meisten derartigen Schnittstellen herstellerspezifisch. Danach setzte sich immer mehr die Verwendung der genormten JTAG-Schnittstelle durch. Diese war ursprünglich für Testfunktionen gedacht (JTAG: Joint Test Action Group), ist aber universell einsetzbar. Mit der JTAG-Schnittstelle können auch mehrere Bausteine angesprochen werden, wobei nur ein Anschluß mit 4 Signalleitungen nach außen benötigt wird. Dies wir als *JTAG-Chain* bezeichnet (Abbildung 2).

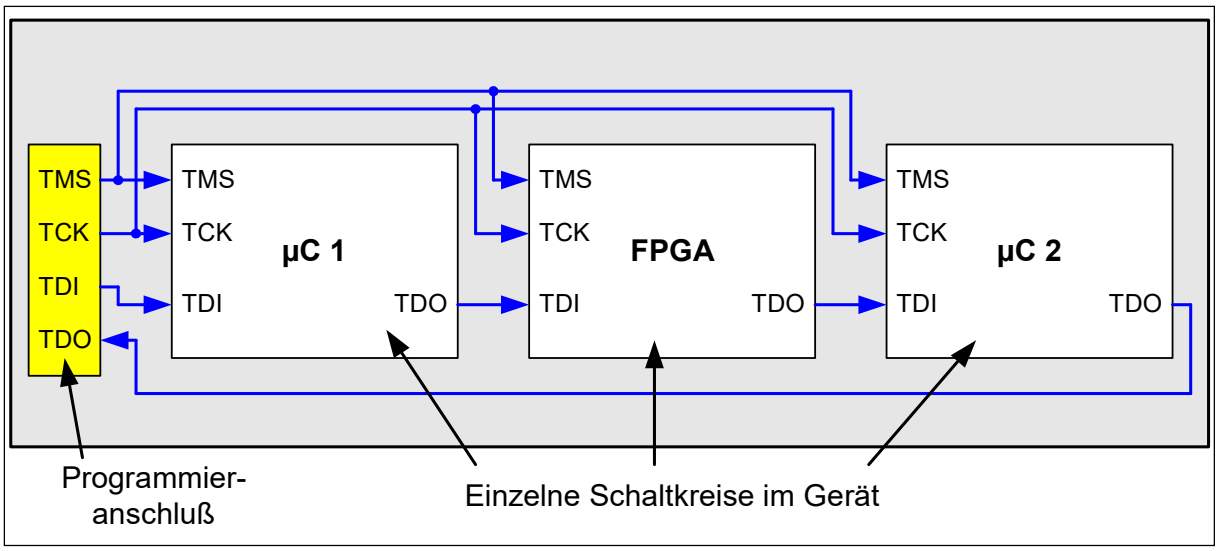

Abbildung 2: JTAG-Chain

Herstellerspezifische ISP-Schnittstellen findet man bei neueren uC kaum noch.
Eine Ausnahme bildet allerdings gerade die Firma ARM. Dort wurde ein proprietärer Ersatz namens SWD (Single Wire Debug) für JTAG entwickelt. Hintergrund ist eine Einsparung von Anschlüssen bei gleichzeitiger Erhöhung der Datenübertragungsrate.
Vorteilhaft an einem ISP-fähigen uC ist, dass der Speicherinhalt auch in eingebautem Zustand veränderbar ist. Man kann damit im *Feld* (d.h. bei ausgelieferten Geräten) noch nachträglich Fehler beseitigen oder neue Funktionen nachrüsten.

3.3 Programmierung im System (Bootloader)

Bei uC ist oft auch die Reprogrammierung des eingebauten Flash durch das eigene Programm möglich. Wenn der uC die Selbstprogrammierung unterstützt, dann kann man ein als *Bootloader* bezeichnetes kleines Programm ständig im Programmspeicher vorhalten.
Die Aufgabe des Bootloaders ist es, sich auf Anforderung über irgendeine beliebige Schnittstelle (z.B. USB) ein neues Anwendungsprogramm zu holen und dies dann in dem freien Teil des Programmspeichers abzulegen. Anschließend wird das neue Programm gestartet und das Gerät nimmt wieder den Normalbetrieb auf.
Der Bootloader bleibt dabei immer im Speicher, er ist im Normalbetrieb nur nicht aktiv. In dem uC befinden sich damit zwei funktional völlig getrennte Programme: Der Bootloader und das Anwendungsprogramm.

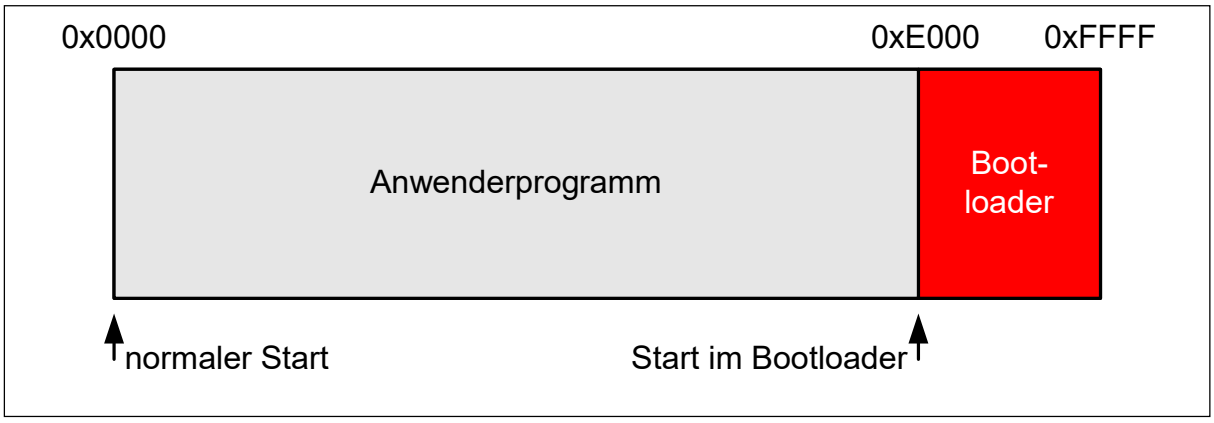

Abbildung 3: Typische Aufteilung des Programmspeichers

Abbildung 3 zeigt eine typische Aufteilung des Programmspeichers in die zwei Teile Bootloader und Anwenderprogramm. Der gesamte verfügbare Programmspeicher umfasst hier 64kB. Davon sind die letzten 8kB für den Bootloader reserviert. Er ist hier nicht nur zur Abgrenzung rot markiert, sondern auch, um eine ebenfalls typische Eigenschaft hervorzuheben: der Bereich, in dem der Bootloader steht, ist oft nicht vom µC selbst programmierbar. Damit wird vermieden, dass eine Amok laufende Anwendung versehentlich den Bootloader löschen oder beschädigen kann. Soll der Bootloaderbereich neu beschrieben werden, dann muss zu einer anderen Methode (z.B. ISP) gegriffen werden.
Bei einem Reset zu Start des Geräts entscheidet häufig ein Spannungspegel an einem Anschluß darüber, ob das Anwenderprogramm (hier Adresse 0x0000) oder der Bootloader (hier 0xE000) ausgeführt wird. Damit kann man mit einem im Gerät angebrachten Schalter in jedem Fall den Bootloader für Diagnose- oder Reparaturzwecke aktivieren. Vor der Auslieferung an den Kunden wird der Schalter dagegen so gestellt, dass immer nur das Anwenderprogramm gestartet wird.

3.4 Vorprogrammierte Bootloader

Zunehmend haben uC bereits einen Speicherbereich mit einem vom Hersteller programmierten Bootloader. Dieser Bootloader ist dann unveränderlich.
Bei dem im Praktikum verwendeten Serie LPC43xx ist ebenfalls ein solcher Bootloader unveränderlich enthalten (UM, Chapter 5).
Es kommt häufig vor, dass sogar zwei Bootloader in einem System enthalten sind. Der erste Bootloader ist dabei der vom Hersteller bereits fest vorprogrammierte Bootloader. Er wird nach der Herstellung des Geräts dazu benutzt, einen zweiten Bootloader in das Flash zu programmieren.
Der zweite Bootloader ist dann vom Gerätehersteller programmiert und kann damit besser auf die jeweilige Anwendung abgestimmt werden.
Der Anwender sieht bestenfalls diesen zweiten Bootloader, beispielsweise wenn ein Update der Anwendung eingespielt werden soll.