

Jacob Borgeson
Stefan Schauer
Horst Diewald
Texas Instruments

Einführung

Entwickler von batteriebetriebenen Bauteilen stehen oft vor der Herausforderung, hochwertige Funktionen und Leistungsdichte zu bieten, wobei gleichzeitig die Batterielebensdauer maximiert werden soll. Anwendungen wie Wasser- oder Gasflussmessgeräte, medizinische Überwachungsgeräte und Remote-Sensoren erfordern oft eine Batterielebensdauer von Monaten oder sogar Jahren. In einigen Fällen müssen Entwickler auch Produkte der nächsten Generation ganz ohne Batterie kreieren, die Energie aus natürlichen Quellen wie Wärme, Vibration oder Licht ziehen. Mit steigender Nachfrage nach längerer Batterielebensdauer und kleineren Batterien in mehr und mehr Anwendungen fordern Benutzer mehr Funktionen und höhere Leistungsdichte ihrer Produkte.

Zur Maximierung der Funktionen und Batterielebensdauer müssen die Entwickler dieser batteriebetriebenen Anwendungen viele Faktoren in ihren Systemarchitekturen und Designs beachten. In diesen Anwendungen ist der Mikrocontroller ein primärer Stromverbraucher. Gleichzeitig müssen Entwickler sorgfältig bedenken, wie Energie genutzt wird. Dieser Artikel beschäftigt sich mit der Aufspaltung der primären Modi, in denen Mikrocontroller Energie verbrauchen. Beschrieben werden kritische Parameter, die in jedem dieser Modi berücksichtigt werden müssen. Außerdem wird ein holistisches Rahmenwerk für Entwickler bereitgestellt, mit dem Mikrocontroller im Kontext bestimmter Anwendungen evaluiert und verglichen werden. Durch das Verständnis der vielen Situationen, in denen Mikrocontroller Energie verbrauchen, können Entwickler Entscheidungen hinsichtlich der Systemarchitektur treffen, optimale Komponenten auswählen und Benutzern von Mikrocontrollern optimierte Funktionen und eine längere Batterielebensdauer bieten.

Benchmarking des Mikrocontroller-Stromverbrauchs für Anwendungen mit extrem geringem Stromverbrauch

Strom ist mehr als nur eine Zahl

Zur Maximierung der Batterielebensdauer müssen Entwickler den Stromverbrauch während der Produktlebensdauer minimieren. Die Gesamtleistung und -energie wird wie folgt definiert:

$$\text{Leistung} = I \times V$$

$$\text{Energie} = I \times \text{Zeit} \times V$$

Zur Minimierung des Strom- oder Energieverbrauchs durch den Mikrocontroller kann ein Entwickler einfach die Produktdatenblätter untersuchen, um den aktuell verbrauchten Strom bei der für die Anwendung notwendigen CPU-Frequenz zu bestimmen. Dieser Stromwert wird mit der Batteriespannung multipliziert, und das Ergebnis wird verwendet, um den Mikrocontroller mit dem geringsten Stromverbrauch auszuwählen. Dies scheint einfach zu sein. Berücksichtigen Sie jedoch einige hypothetische Fragen zu typischen Anwendungen, um zu bestimmen, ob diese Perspektive des Stromverbrauchs umfassend genug ist:

- Wechselt das System in den Standby-Modus, wenn der Mikrocontroller nicht läuft?
- Muss sich das System automatisch nach spezifischen Intervallen selbst wecken?
- Nimmt das System reale Analogsignalmessungen?
- Muss das System Daten zur späteren Analyse oder Übertragung aufzeichnen?

Obwohl dies nur einige wenige Fragen sind, die sich Entwickler bei der Optimierung des Stromverbrauchs stellen müssen, treffen viele der Fragen auf batteriebetriebene Anwendungen zu. Die einfache Methode zur Betrachtung der Strom- und Spannungswerte des Mikrocontrollers führt zu einer ungenauen Darstellung des Stromverbrauchs des Mikrocontrollers.

Damit Entwickler eine umfassende Sicht auf den Stromverbrauch des Mikrocontrollers erhalten, müssen sie vier primäre Stromkategorien beachten:

- **Standby-Strom** – Typische Mikrocontroller-Anwendungen verbringen einen Großteil ihrer Produktlebensdauer in einem niederenergetischen Standby-Modus und warten auf ein internes oder externes Ereignis, um die CPU zur Datenverarbeitung aufzuwecken, Entscheidungen zu treffen und mit anderen Systemkomponenten zu kommunizieren. Bei vielen batteriebetriebenen Anwendungen verbraucht der Standby-Modus die meiste Energie und Batterielebensdauer.
- **Peripherer Strom** – Moderne Mikrocontroller besitzen viele integrierte intelligente Peripheriebauteile, die die Kommunikation mit anderen Systemkomponenten und die Messung von realen Signalen erlauben. In Systemen, die Analogsignale messen, kann dies einen wesentlichen Einfluss auf die Batterielebensdauer haben.

- **Strom zur Datenprotokollierung** – Die meisten Mikrocontroller-Anwendungen protokollieren Daten zur Analyse oder Übertragung zu einem späteren Zeitpunkt. Diese Daten werden mithilfe eines nichtflüchtigen Speichers protokolliert, der sich entweder innerhalb oder außerhalb des Mikrocontrollers befindet. Je nach Frequenz und Menge der Daten, die aufgezeichnet werden müssen, kann die Datenprotokollierung einen großen Einfluss auf die Batterielebensdauer haben.
- **Aktiver Strom** – Die Kenntnis der Strommenge, wenn die CPU aktiv Daten verarbeitet, ist wesentlich, um die Batterielebensdauer zu maximieren.

$$\text{Gesamtenergie} = \text{Energie}_{\text{Aktiv}} + \text{Energie}_{\text{Standby}}$$

$$\text{Zeit} \times I \times V = \text{Zeit}_{\text{Aktiv}} \times I_{\text{Aktiv}} \times V + \text{Zeit}_{\text{Standby}} \times I_{\text{Standby}} \times V$$

Bei batteriebetriebenen Anwendungen sind 3 V die typische Nennbatteriespannung. Außerdem ist es der Spannungswert, der in vielen Datenblättern als Verbrauch angegeben wird. Zur Eingrenzung der Elemente in den Stromberechnungen auf die Elemente, die unter dem Einfluss des Mikrocontrollers stehen, müssen wir die Spannung von weiteren Berechnungen ausschließen, sodass die durchschnittliche Stromstärke im Fokus steht. Natürlich ändern sich der Strom- und Energieverbrauch drastisch, wenn eine andere Spannung angelegt wird. Zusätzlich haben wir die Zeitaspekte außen vor gelassen, sodass jede Stromkomponente als Teil der durchschnittlichen Gesamtstromstärke angesehen wird. Für Informationen zum Verhältnis zwischen der aufbrachten Zeit jeder Komponente und der Stromberechnung müssen Entwickler jedes System einzeln betrachten. In diesem Fall haben wir eine generische Remote-Sensoranwendung gewählt, da sie oft Anwendung findet. Des Weiteren haben wir den MSP430FR59xx als Beispiel-Mikrocontroller mit eingebettetem nicht-flüchtigen ferroelektrischen Speicher (FRAM) zur Datenprotokollierung verwendet.

$$I_{\text{Durchschnitt}} = I_{\text{Standby}} \times \text{Verhältnis}_{\text{Standby}} + I_{\text{Aktiv}} \times \text{Verhältnis}_{\text{Aktiv}} + I_{\text{Peripheriebauteil}} \times \text{Verhältnis}_{\text{Peripheriebauteil}} + I_{\text{Daten}} \times \text{Verhältnis}_{\text{Daten}}$$

Ein Remote-Sensor ist ein Beispiel für ein typisches batteriebetriebenes Produkt mit geringem Stromverbrauch, das häufig in Anwendungen der Industrie, Seismologie, Landwirtschaft, Gebäudeautomatisierung und Sicherheit genutzt wird. Der typische Verarbeitungsfluss in dieser Art von Anwendung sieht wie folgt aus:

Komponente	Anwendungsaktivität
Peripheriebauteil	256 Sensordatenwerte; Druck, Wärme, Vibration, Chemikalien usw.
Aktiv	Verarbeitung der analogen Eingänge, Vergleich der Grenzwerte, Durchschnitt der Eingänge
Daten	Speicherung von 16 Byte im nicht-flüchtigen Speicher pro Zyklus
Standby	Aktiviert sich alle 3 Sekunden, um das Programm auszuführen; Echtzeituhr erhalten, Aufwachfunktion bei E/A-Störung

Tabelle 1: Aktivitätsprofil einer Remote-Sensoranwendung

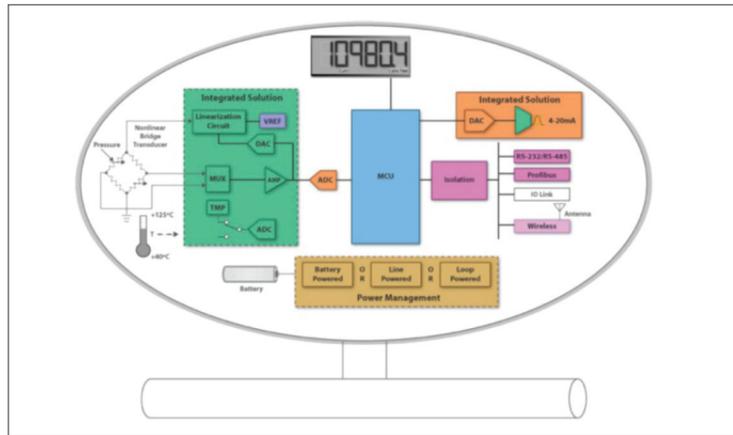


Abbildung 1: Modell einer Sensoranwendung

Strom ist mehr als nur eine Zahl, und die zuvor genannten vier Kategorien bieten ein breites Rahmenwerk, das bei der Entwicklung eines batteriebetriebenen Systems beachtet werden muss. Jede Kategorie in sich ist jedoch komplex, und die spezifischen Anwendungsanforderungen erfordern eine sorgfältige Betrachtung.

Standby-Strom

Entwickler beginnen ihre Prozessorstromanalyse oft damit, den aktiven Verarbeitungsstrom zu betrachten. Obwohl es kontraintuitiv erscheint, ist der Strom, den der nicht betriebene Mikrocontroller verbraucht, oft wichtiger als der aktive Verarbeitungsstrom. Mit Rückblick auf die Remote-Sensoranwendung wacht das System typischerweise alle drei Sekunden aus dem Standby-Modus auf. Also befindet sich das System mehr als 99 % der Gesamtzeit im Standby-Modus. Die nachfolgende Tabelle enthält typische Spezifikationen der „Wolverine“-basierten Mikrocontroller-Reihe MSP-430FR59xx von Texas Instruments (TI) mit 500 nA im Standby-Modus, um den Einfluss des Standby-Stroms auf den Gesamtstrom des Mikrocontrollers in unterschiedlichen Verhältnissen zwischen Aktivzeit und Standby-Zeit darzustellen:

Verhältnis von Aktiv zu Standby	% Zeit im Standby	Zeit(Aktiv) × I(Aktiv) (µAs)	Zeit(Standby) × I(Standby) (µAs)	Gesamtlast (µAs)	% Einfluss von I(Standby) auf Gesamtstrom
1:10	90 %	100	5	107	6,54 %
1:100	99 %	100	50	150	33 %
1:1.000	99,9 %	100	500	600	83,3 %

Tabelle 2: Einfluss von IStandby auf Gesamtstrom basierend auf dem Verhältnis zwischen Aktiv- und Standby-Zeit

Wie Tabelle 2 zeigt, wird der wahre Standby-Strom kritisch, wenn das Verhältnis zwischen Aktiv- und Standby-Zeit 1:100 überschreitet. Bei einem Verhältnis von 1:1.000 beträgt der Standby-Strom zum Beispiel fast 85 % des Systemstroms. Daraus folgt, dass eine Fehlerrate von 10 % in den Standby-Stromschätzungen einem Unterschied von 8,5 % in den Berechnungen der Batterielebensdauer entspricht. Dadurch verringert sich die Batterielebensdauer um Monate oder sogar Jahre.

Um Entwicklern bei der genauen Evaluierung des Standby-Stroms des Mikrocontrollers zu helfen, müssen die folgenden Systemaspekte und Mikrocontroller-Parameter für die Standby-Stromschätzung berücksichtigt werden:

- Automatisches Aufwachen nach Zeitintervallen** – Moderne Mikrocontroller enthalten oftmals Echtzeituhren (RTC – Real-time clock), die in Standby-Modi mit geringem Stromverbrauch laufen und es so dem Mikrocontroller ermöglichen, automatisch nach bestimmten Zeitintervallen aufzuwachen. Die Remote-Sensoranwendung nutzt diese Funktion, um alle drei Sekunden aufzuwachen, um die Analogdaten zu messen. Es ist wichtig, die erforderliche Stromstärke zum Betrieb der RTC im Standby-Modus zu kennen, da dies einen beträchtlichen Teil des Standby-Stroms ausmachen kann.

- **RAM-Erhaltung während Standby** – Die Erhaltung der RAM-Inhalte während des Standbys ermöglicht es Mikrocontrollern, schneller aufzuwachen, ohne den Startcode auszuführen, der wertvolle Energie verbraucht. Dadurch wird Energie und Zeit für eine geringere Systemlatenz gespart. Die zur Aktivierung der RAM-Erhaltungsmodi erforderliche Stromstärke kann signifikant sein und muss sorgfältig berücksichtigt werden.
- **Störfähigkeiten** – Mikrocontroller können im Standby-Modus oft bestimmte Peripheriebauteile aktiv halten, um bei bestimmten Ereignissen wie einem UART-Befehl oder einer GPIO-Störung schneller aufzuwachen. Die Remote-Sensoranwendung überwacht beispielsweise zwei bis drei GPIO-Leitungen gleichzeitig, um den Prozessor zur sofortigen Aktivität aufzuwecken.
- **Stromüberwachung** – Brownout-Reset (BOR) und eine Schaltung zur Spannungsüberwachung (SVS) sind wichtige Schaltungen, die die Integrität der Stromquelle des Mikrocontrollers überwachen. Ausfälle und Störungen der Stromquelle des Mikrocontrollers können die Betriebszuverlässigkeit beeinträchtigen. Daher ist es wichtig, diese Stromstärkewerte in die Standby-Stromschätzungen einzubeziehen. Dabei können den Standby-Werten wenige Nanoampere aber auch bis zu 500 Mikroampere hinzugefügt werden.
- **Temperatur** – Die Temperatur ist eine oft übersehene Variable in Systemdesigns mit geringem Stromverbrauch. Moderne Halbleiterprozesse führen oft zu höheren Ableitsströmen bei höheren Temperaturen. In manchen Fällen beläuft sich dies auf einen 10 bis 15 Mal höheren Standby-Strom zwischen 25 und 85 °C.

Peripherer Strom

Mikrocontroller wie die MSP430FR59xx-Bauteile von TI wurden von Grund auf neu entwickelt, um den niedrigsten Standby-Strom zu optimieren und dabei fortschrittliche Strom- und Clock-Gating-Funktionen, Analogschalt designs für extrem geringen Stromverbrauch und Weiterentwicklungen der Siliziumtechnologie wie eingebetteten FRAM anzuwenden. Im Falle der Remote-Sensoranwendung kann der Mikrocontroller MSP430FR59xx den RTC-Standby-Modus mit RAM-Erhaltung, SVS/BOR und GPIO-Störfunktion mit nur 500 Nanoampere aktivieren.

Bei vielen batteriebetriebenen Anwendungen mit geringem Stromverbrauch kann der Standby-Strom den größten Einfluss auf die Batterielebensdauer haben. Und es ist wichtig, dass Entwickler berücksichtigen, wie die für die Anwendung erforderlichen Funktionen den Standby-Strom beeinflussen. Es kann einen Unterschied von Jahren in der Batterielebensdauer ausmachen oder Systemkosten durch die Verwendung einer geringeren Kapazität und günstigeren Batterie einsparen. Im Beispiel der Remote-Sensoranwendung kann der durchschnittliche Standby-Strom wie folgt geschätzt werden:

$$\text{Durchschn. } I_{\text{Standby}} = I_{\text{Standby}} \times \left(\frac{\text{Zeit}_{\text{Standby}}}{\text{Zeit}_{\text{Gesamt}}} \right) = (2,997 \text{ s} / 3 \text{ s}) \times 0,5 \mu\text{A} = 0,4995 \mu\text{A}$$

(Wobei: 500 nA Energiesparmodus 3,5 mit RTC, SVS und BOR bei 25 °C; SLAS704)

Eingebettete Systeme mit geringem Stromverbrauch verwenden oft analoge oder digitale Schnittstellen, Referenzen und andere Schaltungen, um die Ziele in der Systemfunktion zu erreichen. Bei modernen Mikrocontrollern sind diese Funktionen mehr und mehr integriert, um die Designkomplexität zu vereinfachen, die Systemkosten zu verringern, kleinere Bauteile zu ermöglichen und den Stromverbrauch zu senken. Die Entwickler dieser Systeme müssen sorgfältig sowohl die Funktionen des Mikrocontrollers als auch den Stromverbrauch der Peripheriebauteile berücksichtigen, um die Designs für eine maximale Funktionsweise und Batterielebensdauer zu optimieren. Im Falle der Remote-Sensoranwendung, die zuvor in diesem Artikel vorgestellt wurde, wird ein Analog-Digital-Wandler (ADC) verwendet, um die realen Signale beispielsweise mithilfe von Infrarot-Sensoren, Temperatursensoren oder anderen Sensoren zu messen. Zur genauen Schätzung des peripheren Stroms müssen Entwickler die folgenden Systemdesigneigenschaften und Mikrocontroller-Parameter beachten:

- **Analog-Digital-Wandler** – Der je nach Sampling-Geschwindigkeit und Betriebsmodus verbrauchte Strom; prüfen Sie also das Datenblatt auf den Modus, der der jeweiligen Anwendung am besten entspricht.

- **Komparator** – Komparatoren sind oftmals Lösungen mit geringem Stromverbrauch, die die gleichen grundlegenden analogen Messungen mithilfe eines Analog-Digital-Wandlers erzielen.
- **Spannungsreferenz** – Zur Minimierung der externen Komponenten werden den Mikrocontrollern oft interne Spannungsreferenzen hinzugefügt. Diese Referenzen werden in Analog-Digital-Wandlern, Komparatoren und anderen analogen Schaltungen verwendet. In vielen Fällen ist der von der Spannungsreferenz verbrauchte Strom nicht in den Stromspezifikationen für die genutzten Peripheriebauteile enthalten. Entwickler müssen dies beachten, da die Referenz oft größer sein kann als der Komparator oder der Analog-Digital-Wandler, der die Referenz nutzt.
- **Digitale Schnittstellen** – UART, I2C und SPI gehören zu den vielen in Systemen eingebetteten digitalen Schnittstellen. Jedes dieser Peripheriebauteile verbraucht wertvolle Energie und beeinflusst dadurch die Batterielebensdauer. Bei der Schätzung des Stromverbrauchs müssen Datenraten und Leistungsfähigkeit auf diesen digitalen Schnittstellen eingehend berücksichtigt werden.

In jeder Anwendung, bei der echte Analogsignale gemessen oder digitale Schnittstellen verwendet werden, müssen Entwickler neben dem aktiven Strom oder dem Standby-Strom diese Stromwerte beachten. Dies hängt vom Modus ab, in dem die Peripheriebauteile genutzt werden.

$$\text{Durchschn. } I_{\text{Peripheriebauteil}} = (\text{Zeit}_{\text{Peripheriebauteil}} / \text{Zeit}_{\text{Gesamt}}) \times I_{\text{Peripheriebauteil}} = (0,00128 \text{ s} / 3 \text{ s}) \times 75 \mu\text{A} = 0,032 \mu\text{A}$$

Wobei: 256 Messungen bei 200 kSPS, einer Stromstärke von 75 μA (SLAS704) und 0,00128 s

Strom zur Datenprotokollierung

Viele Mikrocontroller-Anwendungen müssen Messungen und Daten für eine spätere Verwendung aufzeichnen. Zum Beispiel könnten die aktuellsten Daten mit den zuletzt aufgezeichneten Daten verglichen werden, um größere Trends zu erkennen. Die Protokollierung von Daten im Sensor selbst gibt dem Benutzer kritische Informationen im Moment eines Ausfalls. Beispiele dafür sind intelligente Schutzschalter oder die Blackbox in einem Fahrzeug.

Die Datenprotokollierung kann extrem schwer sein, wenn eine kleine, kostengünstige Batterie verwendet wird. Der für die Löschung und Programmierung des Blinkens erforderliche Strom kann zwischen 4 und 12 mA variieren. Diese Stromstärke kann nicht direkt von Knopfzellen geleistet werden, die eine maximale Stromstärke von 3 bis 4 mA bieten. Außerdem dauern derartige Vorgänge länger als 24 Millisekunden, nur um den Speicher vor der Programmierung einzurichten und zu löschen. Der Schreibvorgang im Flash dauert noch einmal etwa genauso lange. Wenn ein externer Speicher verwendet wird, fallen zusätzliche Kosten für die Verwendung der seriellen Kommunikationsschnittstelle zur Datenübertragung an. Dadurch wird das System mit zusätzlichem Peripherie- und Aktivstromverbrauch belastet.

Das ist wichtig, da entweder eine größere und teurere Batterie verwendet oder dem Design ein externer Speicher mit geringerem Stromverbrauch hinzugefügt werden muss. Das sind beides kostspielige Optionen, die sich aufgrund der Größenbeschränkungen oft nur schwer implementieren lassen.

Ein weiterer wichtiger Punkt ist, dass die Datenprotokollierung bei niedrigeren Raten im FRAM den Stromverbrauch drastisch verringert. Dabei kann die Stromstärke oftmals nur 9 μA betragen, um Daten mit einer relativ schnellen Rate von 13 kBit/s zu protokollieren. Dies ist etwa 500 Mal weniger als die Protokollierung mit der gleichen Rate im Flash.

$$\text{Durchschn. } I_{\text{Datenprotokoll.}} = \text{Zeit}_{\text{Daten}} \times (I_{\text{Daten}} / \text{Zeit}_{\text{Gesamt}}) = 0,0012 \text{ s} \times 9 \mu\text{A} / 3 \text{ s} = 0,0036 \mu\text{A}$$

Wobei: 16 Byte bei 13 kBit/s gleich 0,0128 s bei 9 μA entsprechen (Prüfstandsmessung SLAA498)

Aktiver Strom

Aktiver CPU-Strom wird in batteriebetriebenen Anwendungen oftmals als größter Verbraucher angesehen. Wie die vorherigen Abschnitte gezeigt haben, gibt es viele Stromverbraucher in einem Mikrocontroller-System. Deshalb ist es wichtig, dass die Mikrocontroller-Software und -Hardware optimiert sind, um den aktiven Strom zur vorteilhaften Nutzung der intelligenten Peripheriebauteile und niederenergetischen Standby-Strommodi zu minimieren.

Zur genauen Schätzung des aktiven Stroms müssen Entwickler die Nutzung folgender Mikrocontroller-Funktionen, Spezifikationen und Methoden berücksichtigen:

- **Software-Ausführung aus dem nicht-flüchtigen Speicher oder RAM** – Entwickler müssen bei der Schätzung des Stromverbrauchs berücksichtigen, ob Software aus dem nicht-flüchtigen Speicher oder dem RAM ausgeführt wird. Die Ausführung aus dem RAM kann niedrigere Spezifikationen für aktiven Strom bieten. Vielen Anwendungen sind jedoch nicht klein genug, um allein vom RAM ausgeführt zu werden und erfordern, dass Programme aus dem nicht-flüchtigen Speicher ausgeführt werden.
- **Aktivierte oder deaktiverte Bus-Uhren** – Die meisten Mikrocontroller-Anwendungen erfordern während der Software-Ausführung einen Zugriff auf Speicher und Peripheriebauteile. Dies setzt voraus, dass Bus-Uhren aktiviert sind, die ebenfalls in den Schätzungen für den aktiven Strom einbezogen werden müssen.
- **Direkter Speicherzugriff** – Viele Mikrocontroller besitzen Funktionen für den direkten Speicherzugriff, mit denen Peripheriebauteile Aktivitäten durch direkten Zugriff auf den Speicher ausführen können, während die CPU im Ruhemodus ist. So können wesentlich geringere Stromwerte auf gleicher Systemebene erzielt werden. Es muss jedoch dringend geprüft werden, ob dies mit dem jeweiligen Peripheriebauteil oder der jeweiligen Aufgabe möglich ist.
- **Beschleunigung** – Mikrocontroller mit niedrigem Stromverbrauch verwenden oft Beschleuniger, die die Zyklusanzahl und erforderliche Energie für bestimmte Aufgaben verringern. Beispielsweise können bei der AES256-Verschlüsselung bis zu 7.000 CPU-Zyklen ohne Beschleunigung ausgeführt werden. Mit Hardware-Beschleunigern sind bei gleicher Verschlüsselung etwa 500 CPU-Zyklen möglich. In diesem Fall kann die Hardware-Beschleunigung die aktive Verarbeitungszeit um mehr als das 10-Fache reduzieren.
- **Verwendung von optimiertem Code** – Mikrocontroller-Code kann für kürzeste Ausführungszeiten optimiert werden, indem eine intelligente Code-Struktur und Compiler-Optimierungen verwendet werden.

Die Sicherstellung von ordnungsgemäßen Bedingungen und Optimierungsmethoden bei der Schätzung der erforderlichen Zyklusanzahl für die aktiven Verarbeitungsaufgaben und des erforderlichen aktiven Stroms ist wichtig, um die Spitzenströme und den Einfluss auf die Batterielebensdauer exakt abzuschätzen. Mit Blick auf die Remote-Sensoranwendung wird der durchschnittliche Strom wie folgt berechnet:

$$\text{Durchschn. } I_{\text{Aktiv}} = \text{Zeit}_{\text{Aktiv}} \times (I_{\text{Aktiv}} / \text{Zeit}_{\text{Gesamt}}) = 70 \mu\text{s} \times 1230 \mu\text{A} / 3\text{s} = 0,029 \mu\text{A}$$

Wobei: 1.000 Zyklen erforderlich für einen Durchschnitt von 256 Proben bei 16 MHz = 70 μs ;
aktiver Strom bei 16 MHz = 1,23 mA

Zusammenführung: Remote- Sensoranwendungen

Zur Darstellung, wie die Ströme für Standby-Modi, Peripheriebauteile, Datenprotokollierung und aktive CPU zusammen wirken, und der daraus resultierenden Gesamtbatterielevensdauer können die in jedem Abschnitt abgeleiteten Berechnungen für die Remote-Sensoranwendung einfach kombiniert werden, um die voraussichtliche Gesamtbatterielevensdauer für diese typische Anwendung aufzuzeigen. Beginnend mit einer Berechnung des Durchschnittstroms:

$$\begin{aligned} \text{Durchschn. } I_{\text{Gesamt}} &= \text{Durchschn. } I_{\text{Standby}} + \text{Durchschn. } I_{\text{Peripheriebauteil}} + \text{Durchschn. } I_{\text{Aktiv}} + \text{Durchschn. } I_{\text{Datenprotokollierung}} \\ &= 0,4995 + 0,032 + 0,029 + 0,0036 = 0,5641 \mu\text{A} \end{aligned}$$

Wie Sie sehen können, hat der Standby-Strom im Falle einer Remote-Sensoranwendung bei Weitem den größten Einfluss auf den Durchschnittsstrom, gefolgt vom Analog- und Aktivstrom. Der Einfluss auf die einzelnen Komponenten hängt vom Arbeitszyklus der jeweiligen Anwendung ab. Der durchschnittliche Strom kann einfach in die Batterielevensdauer umgerechnet werden, indem die Ladung einer herkömmlichen CR2032-Batterie betrachtet wird, die mit einer Ladung von 230 mAh für weniger als 1 € gekauft werden kann.

$$\begin{aligned} &230.000 \mu\text{Ah} / 0,5641 \mu\text{A} \\ &= 407.729 \text{ Stunden} \\ &= 16.988 \text{ Tage} \\ &= 46 \text{ Jahre Batterielevensdauer (viel länger als die Selbstentladungszeit von 8–10 Jahren)} \end{aligned}$$

Da der Stromverbrauch des Mikrocontrollers minimiert wurde, hat der Entwickler die Möglichkeit, dem System weitere Systemfunktionen hinzuzufügen und dennoch eine Batterielevensdauer einer einzigen Knopfzelle von 5 bis 10 Jahren zu erzielen. Zusätzlich lässt sich diese Berechnung einfach erweitern, indem der gleiche Fall verwendet wird, um mehrere verschiedene Mikrocontroller zu evaluieren und so zu bestimmen, welcher den geringsten Stromverbrauch für eine spezifische Anwendung bietet. In der nachfolgenden Tabelle werden vier unterschiedliche Mikrocontroller im gleichen Beispiel einer Remote-Sensoranwendung bewertet:

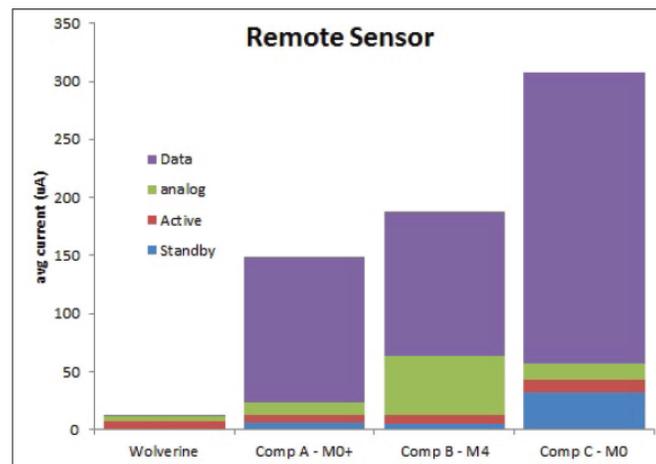


Abbildung 2: Vergleich des Durchschnittstroms

Die als Beispiel verwendete Remote-Sensoranwendung ist ein typischer Anwendungsfall für viele batteriebetriebene Produkte mit geringem Stromverbrauch, die eine lange Batterielevensdauer erfordern. Wasser- und Gasmessgeräte, Herzfrequenzmonitore, Schrittzähler und industrielle Sensoren wenden alle ähnliche Techniken an, um Kosten zu minimieren, behördliche Standards zu erfüllen und Kundenanforderungen gerecht zu werden. Zur Maximierung der Batterielevensdauer müssen alle vier Aspekte des Mikrocontroller-Stromverbrauchs berücksichtigt werden.

Zusammenfassung

Für Entwickler ist es eine enorme Herausforderung, die Bedürfnisse einer erhöhten Systemfunktion und Leistungsdichteziele zu erfüllen, während die Batterielebensdauer des Produkts verlängert werden soll. Zur effektiven Entwicklung von Produkten mit möglichst langer Batterielebensdauer oder sogar ohne Batteriebetrieb ist es erforderlich, sowohl die Systemanforderungen als auch die Stromspezifikationen des Mikrocontrollers zu kennen. Dies ist viel komplexer, als einfach zu schätzen, wie viel Strom von der aktiven CPU verbraucht wird. Je nach entwickelter Anwendung könnte der Strom für Standby-Modi, Peripheriebauteile oder Datenprotokollierung einen größeren Einfluss auf die Batterielebensdauer haben als der CPU-Strom.

Dieser Artikel wurde verfasst, um ein einfaches Rahmenwerk für Entwickler bereitzustellen, damit diese den Stromverbrauch von Mikrocontrollern, die Stromoptimierung und den Vergleich des Stromverbrauchs verschiedener Mikrocontroller-Lösungen verstehen und so die spezifischen Anforderungen ihrer Anwendung erfüllen.

TI wird weiterhin die Grenzen für geringen Stromverbrauch verschieben, indem ALLE wichtigen Bereiche des Stromverbrauchs optimiert werden. Das MSP430™-Portfolio mit extrem geringem Stromverbrauch von TI wurde auf der Grundlage eines geringen Stromverbrauchs entwickelt, indem Weiterentwicklungen der Siliziumtechnologie, Designtechnologien mit geringem Stromverbrauch, intelligente Peripheriebauteile und Beschleuniger sowie Designtools mit geringem Stromverbrauch verwendet werden.

Weitere Informationen zu Designs mit extrem geringem Stromverbrauch und das umfassende TI-Portfolio aus Produkten mit extrem geringem Stromverbrauch finden Sie unter www.ti.com/ulp.

Important Notice: The products and services of Texas Instruments Incorporated and its subsidiaries described herein are sold subject to TI's standard terms and conditions of sale. Customers are advised to obtain the most current and complete information about TI products and services before placing orders. TI assumes no liability for applications assistance, customer's applications or product designs, software performance, or infringement of patents. The publication of information regarding any other company's products or services does not constitute TI's approval, warranty or endorsement thereof.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers' products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers' products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have **not** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products

Audio	www.ti.com/audio
Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DLP® Products	www.dlp.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
OMAP Applications Processors	www.ti.com/omap
Wireless Connectivity	www.ti.com/wirelessconnectivity

Applications

Automotive and Transportation	www.ti.com/automotive
Communications and Telecom	www.ti.com/communications
Computers and Peripherals	www.ti.com/computers
Consumer Electronics	www.ti.com/consumer-apps
Energy and Lighting	www.ti.com/energy
Industrial	www.ti.com/industrial
Medical	www.ti.com/medical
Security	www.ti.com/security
Space, Avionics and Defense	www.ti.com/space-avionics-defense
Video and Imaging	www.ti.com/video

TI E2E Community

e2e.ti.com