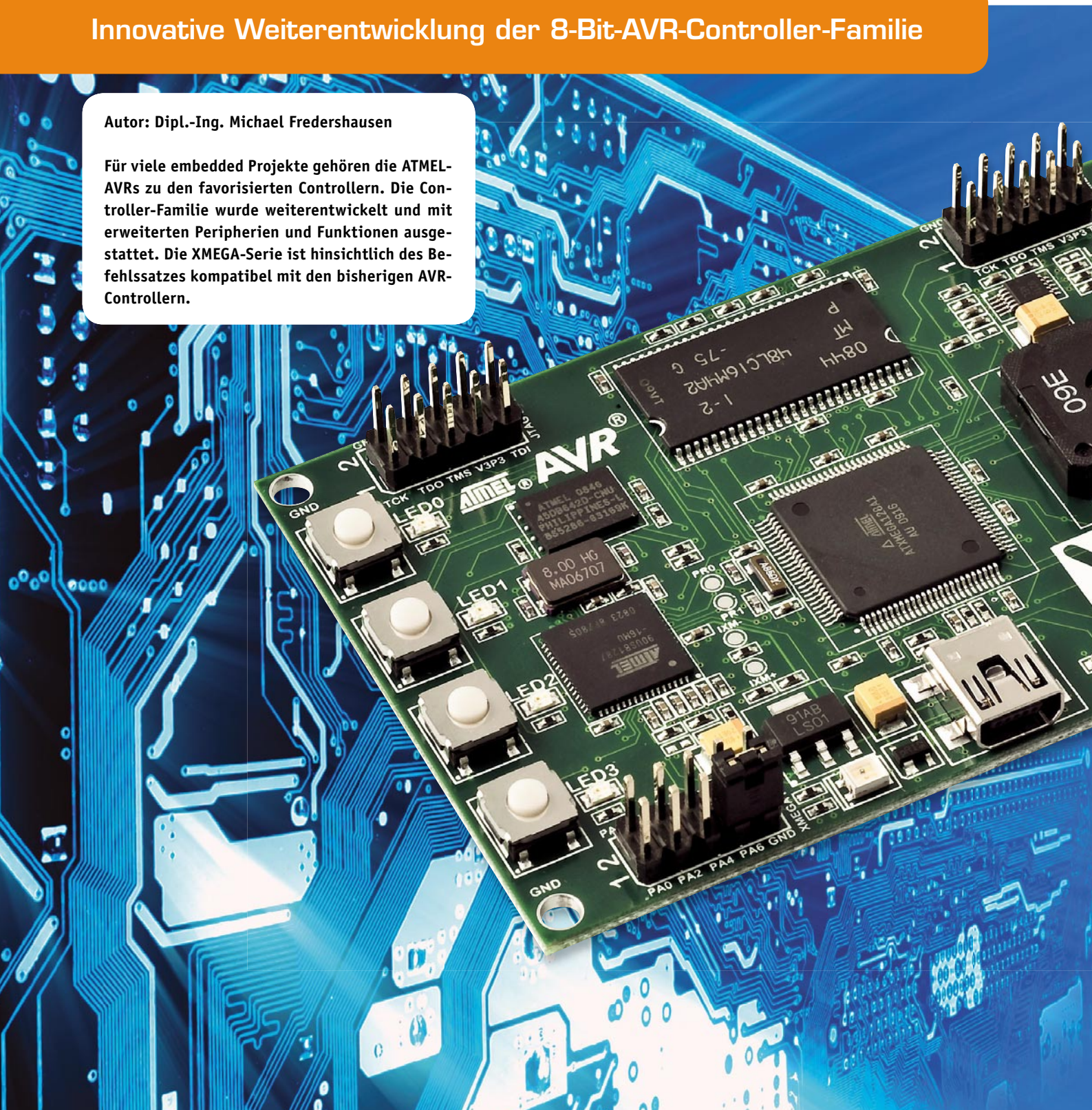
Vom AVR zu XMEGA

Innovative Weiterentwicklung der 8-Bit-AVR-Controller-Familie

Autor: Dipl.-Ing. Michael Fredershausen

Für viele embedded Projekte gehören die ATMEL-AVRs zu den favorisierten Controllern. Die Controller-Familie wurde weiterentwickelt und mit erweiterten Peripherien und Funktionen ausgestattet. Die XMEGA-Serie ist hinsichtlich des Befehlssatzes kompatibel mit den bisherigen AVR-Controllern.



Allgemeines

Mittlerweile ist aus keinem modernen Produkt der Konsum- und Industrieelektronik der Einsatz von hochintegrierten Mikrocontrollern mehr wegzudenken.

In erster Line werden hier in-Circuit-programmierbare Flashcontroller mit hoher Peripheriedichte eingesetzt, die nur ein Minimum an externer Zusatzbeschaltung benötigen. Der Einsatz solcher flashbasierender Mikrocontroller mit hoher Peripheriedichte verringert die Produktionskosten und verbessert somit die Akzeptanz und Wettbewerbsfähigkeit des Gesamtproduktes am Markt.

Eine herausragende Vorreiterrolle bei der Entwicklung von in-Circuit-programmierbaren Flashcontrollern nimmt der US-amerikanische Halbleiterhersteller ATMEL ein.

ATMEL hat schon 1990 mit der Entwicklung der legendären 8-Bit-AVR-Controller-Familie begonnen. Inzwischen ist der AVR-Controller zum Industriestandard geworden und hat den etwas in die Jahre gekommenen 8051-Controller aufgrund der hohen Verarbeitungsgeschwindigkeit und seiner modernen Prozessorarchitektur als Standard abgelöst.

2008 hat ATMEL die innovative Weiterentwicklung der AVR-Controller begonnen und die auf dem gleichen CPU-Core basierende XMEGA-Familie entwickelt. Bei der Entwicklung der XMEGA-Controller wurde erstmals ein besonders stromsparender CMOS-Prozess, die sogenannte Pico-Power-Technologie, eingesetzt, da die Industrie immer mehr auf batteriegestützte Produkte mit extrem niedrigem Stromverbrauch setzt. Als notwendige Konsequenz hat ATMEL nun auch damit begonnen, die herausragenden Eigenschaften im Stromverbrauch, die durch Einsatz der Pico-Power-Technologie erreicht werden, auf die Standard-AVR-Controller zu übertragen. Es ist eine Vielzahl von AVR-Controllern mit dem Suffix P, PA verfügbar (siehe hierzu Tabelle 1) und weitere werden noch folgen.

Hierbei steht das Suffix P für die Pico-Power-Technologie, bei der im Power-down-Mode die typische Stromaufnahme bei <100 nA liegt. Zum Vergleich benötigen die Standard-AVRs, wie z. B. der ATmega16, im selben Mode circa 1 µA.

Der weitere Zusatz A sagt aus, dass AVR-Controller mit diesem Suffix auch im aktiven Modus circa 20 % weniger Strom verbrauchen als entsprechende AVR-Controller ohne diesen Zusatz.

Dem „ELVjournal"-Leser sind die Standard-AVR-Controller und seine P-Derivate bestens bekannt, da diese aufgrund von Rechengeschwindigkeit, Peripherieausstattung und niedrigem Stromverbrauch in sehr vielen ELV-Produkten Verwendung finden.

Eine Vielzahl von AVR-Derivaten, Tiny AVRs, ATmegas und neuen XMEGA-Controllern ist inzwischen verfügbar. Devicelisten finden Sie unter www.atmel.com. Alle AVR-Controller basieren auf demselben CPU-Kern (AVR-Core), wobei die höchste Peripheriedichte bei den neuen XMEGA-Controllern realisiert wurde. Bild 1 illustriert die Peripheriedichte der AVR-Controller im Vergleich.

Die Low-Pin-MCUs (8–32 Pins) werden als Tiny Controller bezeichnet. Sie sind mit max. 8-K-Flash-Programmspeicher ausgestattet und besitzen gegenüber den mit max. 100 Pins ausgestatteten ATmega-AVRs reduzierte Peripheriefunktionen sowie kleinere SRAM- und Flashspeicher.

Auch die neuen Controller der XMEGA-Familie sind mit dem vom Standard-AVR bekannten MCU-Core ausgestattet, mit dem Vorteil, dass dieser Core mit 32 MHz deutlich höher getaktet werden kann und somit auch eine höhere Rechenleistung zur Verfügung steht. Bekanntermaßen können die Standard-AVR-

Tabelle 1

Device	Flash (KBytes)	EEPROM (Bytes)	SRAM (Bytes)	I/O Pins	F. max (MHz)	Vcc (V)
ATmega1284P	128	4096	16 K	32	20	1,8–5,5
ATmega164A	16	512	1024	32	20	1,8–5,5
ATmega164PA	16	512	1024	32	20	1,8–5,5
ATmega165PA	16	–	1024	54	16	1,8–5,5
ATmega168A	16	512	1024	23	20	1,8–5,5
ATmega168PA	16	512	1024	23	20	1,8–5,5
ATmega169PA	16	512	1024	54	16	1,8–5,5
ATmega324PA	32	1024	2048	32	20	1,8–5,5
ATmega3250P	32	1024	2048	69	20	1,8–5,5
ATmega325P	32	1024	2048	54	20	1,8–5,5
ATmega3290P	32	1024	2048	69	20	1,8–5,5
ATmega329P	32	1024	2048	54	20	1,8–5,5
ATmega48PA	4	256	512	23	20	1,8–5,5
ATmega644PA	64	2048	4096	32	20	1,8–5,5
ATmega88PA	8	512	1024	23	20	1,8–5,5

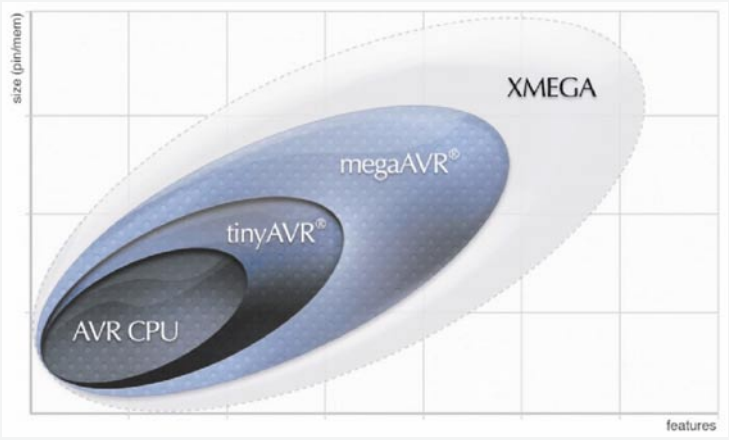


Bild 1: Peripheriedichte der AVR-Controller im Vergleich

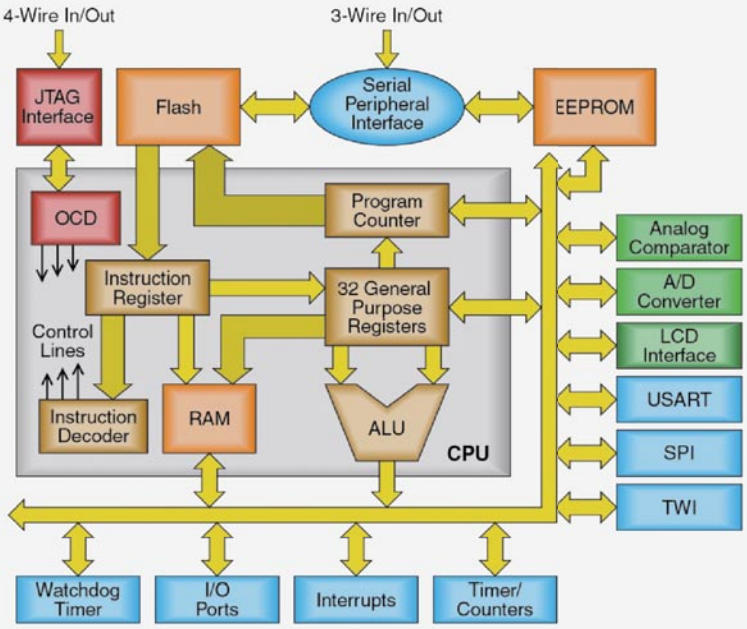


Bild 2: AVR-Prozessor-Architektur

ATmega-Controller wie z. B. der ATmega88P mit max. 20 MHz getaktet werden. Bei MCUs der XMEGA-Familie steht das Präfix X für erweiterte, „eXtended" Peripherie- und Prozessor-Features.

Für alle AVR-Controller gilt aber aufgrund des gleichen Cores, dass diese untereinander 100 % softwarekompatibel sind. Dies bedeutet, dass alle AVR- und XMEGA-Controller mit denselben Softwarewerkzeugen, sprich demselben Assembler und C-Compiler, programmiert werden können.

Hierbei ist in erster Linie auf das von ATMEL frei zur Verfügung gestellte AVR-Studio mit integriertem GNU-Compiler zu verweisen, das in seiner MCU-Deviceliste auch die schon verfügbaren XMEGA-Controller aufführt. Aktuelle Versionen des AVR-Studio-Softwarepaketes können auf der ATMEL-Homepage (www.atmel.com) heruntergeladen werden.

Auch die vom Standard-AVR-Controller bekannten Hardware-Tools können weiterhin zur Controllerprogrammierung und zum Debugging mit den XMEGA-Controllern eingesetzt werden.

Dies ist ein wesentlicher Vorteil für den XMEGA-Entwickler, da er sich beim Umstieg auf den XMEGA-Controller der gleichen Soft- und Hardwarewerkzeuge bedienen kann. Bevor in späteren Ausführungen detailliert auf die erweiterten Peripheriefunktionen und Anwendervorteile der XMEGA-Controller eingegangen wird, sollen die Architektur des AVR-Cores sowie seine besondere Eignung zur Hochsprachenprogrammierung in C erläutert werden.

AVR-Core – Prozessor-Architektur, Registermodell

Alle ATMEL-AVR-Controller basieren mit ihrem AVR-Core auf einer erweiterten RISC-(Reduced-Instruction-Set-Computer-)Architektur. Die AVR-Prozessor-Architektur sowie das Registermodell sind dem Blockdiagramm in Bild 2 zu entnehmen. Die im Blockschaltdiagramm aufgeführten Peripherieelemente entsprechen den AVR-ATmega-Controllern. Alle „ATmegas" sind mit in-Circuit-programmierbarem Flashspeicher sowie inhärenten SRAM und EEPROM ausgestattet und verfügen in der Regel über folgende Peripherieelemente:

- 10-Bit-ADC
- Analog-Komparator
- 1x 16-Bit-Timer, 2x 8-Bit-Timer
- USART TWI (I^2C), SPI-Schnittstelle (LCD-Controller optional)
- Watchdog-Unit
- JTAG-Interface

AVR-Core-Details

Der 8-Bit-AVR-Core nutzt die bekannte Harvard-Struktur, in der Programm- (Flash) und Datenspeicher (SRAM) voneinander getrennt ausgeführt worden sind. Der Zugriff auf den inhärenten Programmspeicher erfolgt äußerst zeiteffektiv unter Nutzung des sogenannten „single level pipelining". Während vom Prozessor eine Instruktion (Befehl) ausgeführt wird, wird schon die nächste Instruktion aus dem Programmspeicher geladen (prefetched).

Zentrales Element der AVR-Architektur ist der Fast-Access-Registerfile-Block, der mit 32x 8-Bit-Arbeitsregister ausgestattet ist und direkt mit der ALU (Arithmetic Logic Unit) verbunden ist. Die Register dieser Registerblocks können besonders effektiv als Akkumulator- und Pointerregister (max. drei 16/24 Memory Pointer) genutzt werden.

Der AVR-Controller wird auch als sogenannte One-Cycle-Maschine oder Single-Cycle-Core bezeichnet, der eine max. Verarbeitungsgeschwindigkeit 1 MIPS/MHz zulässt. Während nur eines Clock-Cycles ist der Prozessorkern in der Lage, zwei Operanden aus dem Registerblock in die ALU (Arithmetic Logical Unit) zu laden, eine gewünschte arithmetische Operation auszuführen und das Resultat anschließend in eines der 32 Arbeitsregister zurückzuspeichern. Neben dieser hohen Verarbeitungsgeschwindigkeit zeichnet den AVR-Kern auch seine hohe Codedichte (Code Density) aus. Diese wird durch die Verwendung von auf Hochsprachenprogrammierung abgestimmten Instruktionen erreicht. Somit kann auch bei Programmierung in C ein kompakter Programmcode erzeugt werden.

AVR- und XMEGA-Controller – optimiert für die C-Programmierung

Die Entwicklung des AVR-Cores wurde in enger Zusammenarbeit mit dem namhaften Compiler-Hersteller IAR

Tabelle 2

C-Programmbeispiel

```
void example (void)
{
long x1,x2;
int x3;
if (x1!=x2) x3+=5;
...
}
```

Tabelle 3

```
        Nutzung des Registerblocks (R0-R31)

        Variable x1 =>R3-R0
        Variable x2 =>R7-R4
        Variable x3 =>R17:R16

CP      R0,R4; x1-x2 (byte 0)
CPC     R1,R5; x1-x2-Carry(byte 1)
CPC     R2,R6; x1-x2-Carry(byte 2)
CPC     R3,R7; x1-x2-Carry(byte 3)
        BRE EQUAL; Branch if equal
        SUBI R16,LOW (0xFFFB); x3+5 low byte / 0xFFFA ->2 Complement 5
        SUBI R17,HIGH(0xFFFB); x3+5 high byte
```

Generierter Assemblercode

durchgeführt. Mittels Benchmarking wurden besonders für die C-Programmierung effektive Assemblerbefehle ermittelt, die dann in die AVR-Controller übernommen wurden. Viele bekannte Mikrocontroller-Architekturen benutzen nur 1 oder 2 Akkumulator-Register. Erinnern wir uns an den 8051, so ist dieser nur mit maximal 2 Akkus (A, B) ausgestattet, was zu geringen Ausführungszeiten und geringer Codedichte führt, da eine Vielzahl von Operationen über den Flaschenhals Akkumulator abgewickelt werden müssen.

Ein weiterer entscheidender Vorteil des AVR-Cores sind die 32 8-Bit-Arbeitsregister. Bei der Programmierung in der Hochsprache C ist es sinnvoll, lokale Variablen zu nutzen, da diese nur temporär das interne SRAM nutzen. Um lokale Variablen schnell und codeeffizient an C-Prozeduren zu übergeben, wird in erster Linie der AVR-Registerblock eingesetzt.

Das in Tabelle 2 ersichtliche einfache C-Programm wird unter Nutzung C-optimierter Assemblerbefehle sowie des AVR-32x-Registerblocks in einen äußerst effektiven AVR-Assemblercode umgesetzt.

Erläuterungen zum Programmcode (Tabelle 2, Tabelle 3)

Im C-Programm gemäß Tabelle 2 werden die Variablen X1, X2 als 32-Bit-Variablen (long) definiert und benötigen somit 4 Bytes. Die Integer-Variable X3 hingegen benötigt nur 2 Bytes. Alle drei Variablen sind als „local" definiert. Wenn Variable X1 ungleich X2 ist, soll zur Integer-Variable X2 der Wert 5 addiert werden.

Der zugehörige Assemblercode in Tabelle 3 zeigt die effektive Nutzung spezieller Assemblerbefehle sowie die optimale Nutzung des AVR-Registerblocks.

Viele der am Markt verfügbaren 8-Bit-MCUs benötigen weitaus mehr Assembler-Instruktionen als der in Tabelle 3 abgebildete AVR-Assemblercode.

Unter Nutzung des CPC-Befehls (compare with carry) wird der schnelle Vergleich von 16- und 32-Bit-Variablen ermöglicht.

Die Addition des Wertes 5 zur Integer-Variablen X3 wird durch Verwendung des SUBI-Befehls (SUB immediate) ausgeführt, da eine Subtraktion des 2er-Komplements der Addition entspricht.

XMEGA-Controller – die Weiterentwicklung der AVRs

Bedingt durch die hohe Marktakzeptanz der 8-Bit-AVR-Controller haben sich die ATMEL-Design-Ingenieure und Produktmanager die Frage gestellt, ob sich durch erweiterte, intelligente Peripheriefunktionalität bei gleichem Prozessorcore und ähnlicher Kostenstruktur die Marktverbreitung von AVR-Derivaten noch weiter steigern lässt.

Die Antwort haben die ATMEL-Chip-Designer mit der Entwicklung der neuen leistungsstarken XMEGA-Familie gegeben. Aufgrund der nachfolgend aufgeführten Prozessor-Features werden die „XMEGAs" den steigenden Anforderungen von komplexen „embedded designs" gerecht.

Aufgrund der umfangreichen Peripheriedichte der XMEGA-Familie hat sich ATMEL dazu entschieden, drei XMEGA-Sub-Familien zu realisieren. Dies hat den entscheidenden Vorteil, dass nun der Anwender entsprechend seinen Peripherie- und Speicheranforderungen den geeigneten Controller aus den A1-, A3- und A4-Familien auswählen kann. Eine Sonderstellung in der Nomenklatur nehmen die sogenannten D-Derivate ein, die es zu jeder der drei Sub-Familien gibt. Auf die D-Derivate wird gesondert eingegangen.

XMEGA-Highlights in der Kurzübersicht

Nachfolgend werden die wesentlichen Controllermerkmale der XMEGA-Prozessorfamilie aufgeführt. Absolute Neuheiten des XMEGA-Designs sind das innovative Eventsystem sowie der 4-kanalige DMA-Controller.

Des Weiteren sind die integrierten 12-Bit-ADC- und 12-Bit-DA-Wandler sowie der extrem niedrige Stromverbrauch, der durch Einsatz der schon erwähnten Pico-Power-Technologie erreicht wird, hervorzuheben.

XMEGA-Features im Überblick

- Pico-Power™-Technologie, 100 nA im Power-down-Mode
- Versorgungsspannung: 1,6–3,6 V
- Max. Prozessor-Clock: 32 MHz (max. 32 MIPS)
- Flexibles PLL-programmierbares Clocksystem
- Interne Oszillatoren; 32 MHz, 2 MHz, 32 kHz, 32 kHz Ultra Low Power
- Programmspeicher Flash 16–256 K Flash, 2–16 K SRAM, 2–8 K EEPROM
- Separates Bootflash 4–8 K
- Programmierbarer Interruptcontroller mit programmierbaren Interruptprioritäten
- Externes Bus-Interface (EBI) bei Controllern der A1-Familie
- Eventsystem mit 4-kanaligem DMA-Controller
- Mehrkanalige High-Speed-12-Bit-ADCs und -12-Bit-DACs
- AES DES Crypto engine Hardware CRC Generator
- Durchgängige Verwendung von mehreren 16-Bit-Timern/-Countern
- Max. 8 UARTS, max. 8 SPI, max. 4 TWI (I^2C), IRDA

Der erste Überblick zeigt schon eindrucksvoll die Leistungsfähigkeit dieser neuen Mikrocontroller-Familien. Auf die aufgelisteten Features wird im zweiten Teil im nächsten Heft detailliert eingegangen. ELV