Kurze Einführung zu Digitalen Signalprozessoren

Inhalt

Die folgende Einführung will Ihnen den Einstieg in die Laborübung Digitale Signalverarbeitung II erleichtern, indem Sie eine kurze Einführung zu folgenden Aspekten gibt:

- zu digitalen Signalprozessoren (Englisch: Digital Signal Processors – DSPs),
- dem in der Laborübung verwendeten Evaluationsboard (EVB),
- dem in der Laborübung verwendeten DSP: Blackfin BF-561 der Firma Analog Devices.

Weitere Details zur Hardware und zur Programmierung des BF-561 finden Sie in den Manuals von Analog Devices.

1 Digitale Signalprozessoren

Digitale Signalprozessoren (DSPs) sind Mikroprozessoren, die speziell auf rechenintensive Aufgaben der digitalen Signalverarbeitung zugeschnitten sind. Wichtige Beispiele für solche Aufgaben sind:

- FIR-Filterung (diskrete Faltung)
- IIR-Filterung
- FFT (Fast Fourier Transform)
- Berechnung von Korrelationsfunktionen

Bei all diesen Rechenaufgaben kommen vorwiegend Multiplikationen und Additionen vor. Deshalb ist sowohl die Hardware-Architektur als auch der Befehlssatz von DSPs für eine schnelle Berechnung von Multiplikationen und Additionen optimiert.

1.1 Hardware Architekturen

1.1.1 Von Neumann Architektur

Bei der von Neumann Architektur werden der Programmcode und die Daten in einem gemeinsamen Speicher abgelegt. Das heißt, Datenspeicher (Data Memory DM) und Programmspeicher (Program Memory PM) bilden eine Einheit, die mit dem Befehlsrechner und dem Datenrechner typischerweise über einen gemeinsamen Bus verbunden ist. Der Befehlsrechner (Instruction Processor IP) liest die Befehle aus dem Speicher und dekodiert sie. Die Ausführung des Befehls übernimmt dann der Datenrechner (Data Processor DP), der dafür über eine Arithmetic Logic Unit (ALU) verfügt. Zur Ausführung der Befehle ist es meist notwendig, Operanden aus dem Speicher zu holen, wobei für jeden Speicherzugriff ein Taktzyklus benötigt wird. Entscheidend ist, dass bei der von Neumann Architektur ein gleichzeitiger Zugriff auf den Programmcode und die Daten nicht möglich ist. Für einen Multiplikationsbefehl sind beispielsweise drei Speicherzugriffe notwendig: einer zum Lesen des Befehls und je einer zum Lesen der beiden Faktoren. Deshalb sind bei einer von Neumann Architektur allein für die Speicherzugriffe eines Multiplikationsbefehls mindestens drei Taktzyklen notwendig. Da Signalprozessoren jedoch gerade Multiplikationen sehr schnell ausführen sollen, ist die von Neumann Architektur für DSPs nicht geeignet.

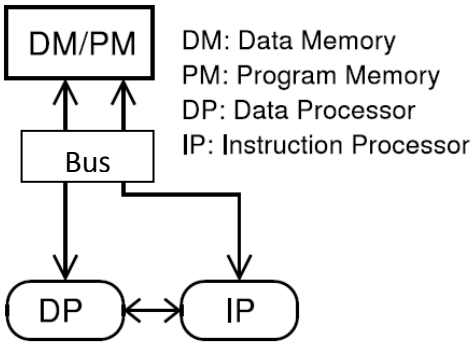

Abbildung 1: von Neumann Architektur [2].

Vorteile der von Neumann Architektur:

- relativ geringer Hardwareaufwand
- keine Dateninkohärenzen
- der Programmcode kann während der Laufzeit verändert werden

1.1.2 Harvard-Architektur

Bei der Harvard-Architektur sind Programmspeicher und Datenspeicher getrennt und über zwei separate Bus-Systeme mit Befehlsrechner und Datenrechner verbunden. Somit können in einem Taktzyklus sowohl ein Befehl aus dem Programmspeicher als auch ein Operand aus dem Datenspeicher geholt werden. Weil viele arithmetische Operationen zwei Operanden benötigen (z.B. Addition und Multiplikation), werden manchmal statt einem Datenspeicher und einem Datenbus zwei Datenspeicher und zwei Datenbusse verwendet, so dass ein Befehl und zwei Operanden in einem Taktzyklus gelesen werden können. Der gleichzeitige Zugriff auf Programmcode und Daten führt zu einem enormen Geschwindigkeitsvorteil bei rechenintensiven Aufgaben. Allerdings bedeuten die getrennten Speicher und Bus-Systeme auch einen erhöhten Hardware-Aufwand. Die klassische Harvard-Architektur und Modifikationen davon wurden in vielen älteren DSPs verwendet.

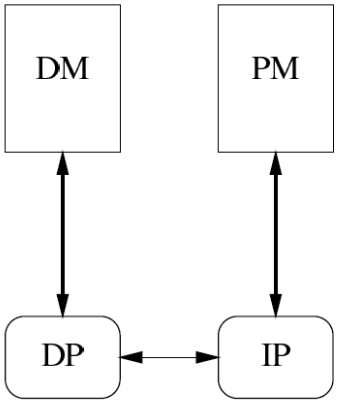

Abbildung 2: Harvard-Architektur [2].

Vorteile der Harvard-Architektur

- höhere Verarbeitungsgeschwindigkeit
- Programmierfehler führen nicht so leicht zum Überschreiben des Programmcodes

1.1.3 Modifizierte Harvard-Architektur

Die meisten modernen DSPs besitzen eine modifizierte Harvard-Architektur, bei welcher der Speicher in unterschiedliche Hierarchien aufgeteilt ist, z.B. in einen Level 1 (L1) und Level 2 (L2) Cache. Dabei ist nur die unterste Ebene (L1 Cache) getrennt in Programmspeicher und einen oder zwei Datenspeicher. Auf diese Weise können viele der Vorteile einer von Neumann Architektur und einer klassischen Harvard-Architektur vereint werden.

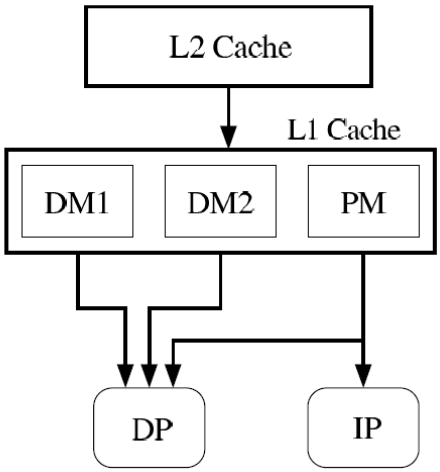

Abbildung 3: Modifizierte Harvard-Architektur [1].

2 Evaluationsboard

Die Laborübung wird mit dem Evaluationsboard ADSP-BF561-EZLITE durchgeführt, welches im Folgenden näher beschrieben wird.

2.1 Aufbau

Das Evaluationsboard enthält neben dem DSP noch die Hardware für Ein-/Ausgabe und den Programmieranschluss. Die für uns wichtigsten Komponenten des Evaluationsboards sind:

- Der Digitale Signalprozessor: ADSP-BF561 Blackfin von Analog Devices
- Speicherbausteine: 64 MB SDRAM und 8 MB Flash Memory
- Audio-Codec AD1836 A zur Digitalisierung von bis zu vier analogen Audioeingängen und zur Ausgabe von bis zu sechs analogen Audioausgängen
- 16 LEDs, die über I/O-Pins des DSPs, die sogenannten Programmable Flags (PFs), angesteuert werden können
- 4 Mini-Taster, die als Eingabe für den DSP dienen

Das folgende Blockschaltbild zeigt die wichtigsten Komponenten des Evaluationsboards und deren Verbindungen untereinander.

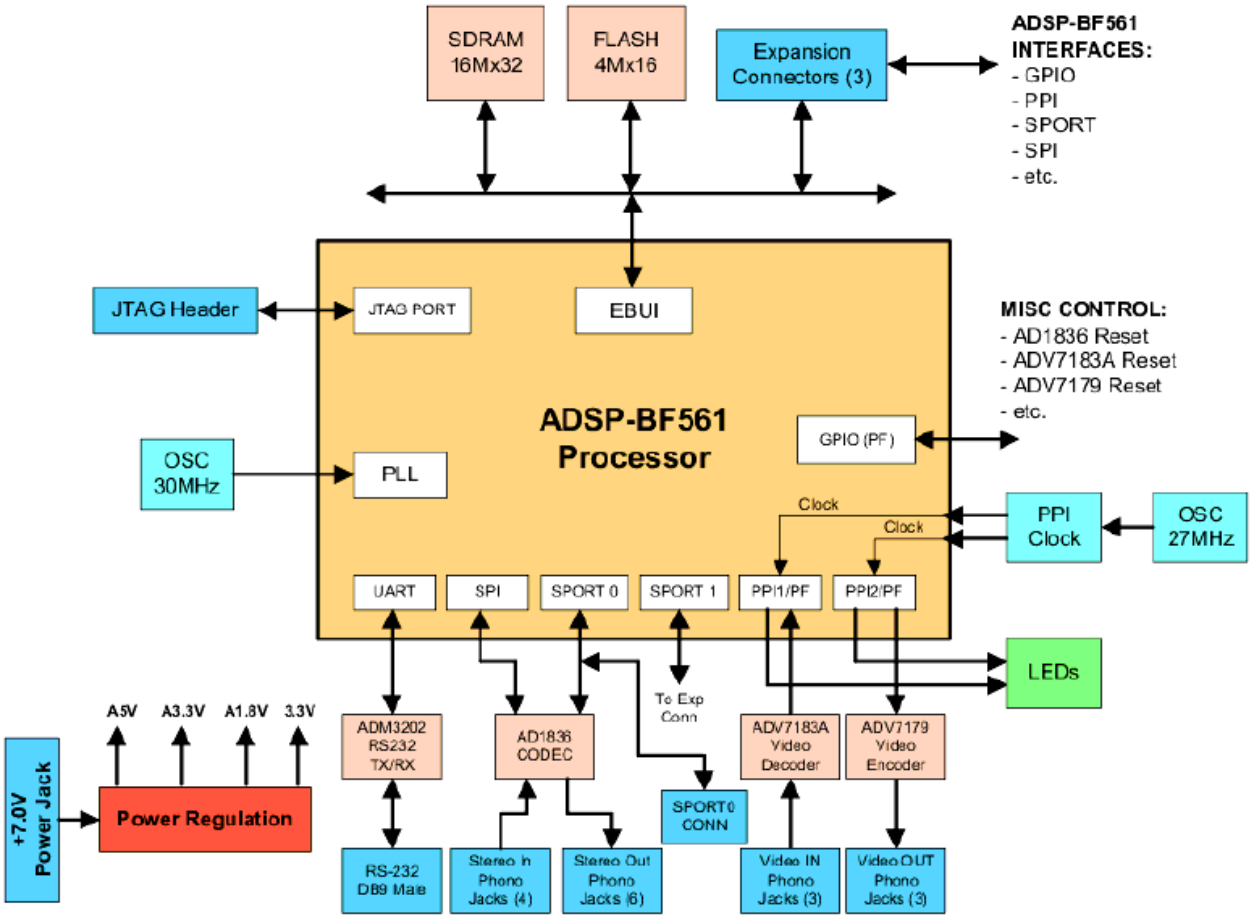

Abbildung 4: Blockschaltbild des Evaluationsboards [3]

2.2 Audio Codec

Der Audio Codec verfügt über zwei Stereo Eingänge (4 Kanäle) und drei Stereo Ausgänge (6 Kanäle). Er ist über den SPORT0 mit dem DSP verbunden. Die analogen Signale werden über Cinch-Buchsen abgegriffen/bereitgestellt.

2.2.1 Wichtigste Kenndaten der AD-Wandler/DA-Wandler

- Abtastrate: bis zu 96 kHz
- Auflösung: bis zu 24 Bit
- Aussteuerbereich der ADCs: 2,18 V (Effektivwert), entspricht 6,17 V Spitze – Spitze
- Maximale Ausgangsspannung DACs: 2,00 V (Effektivwert), entspricht 5,65 V Spitze – Spitze

2.2.2 Konfiguration des Codecs

Der Audio Codec kann durch den DSP über die SPI (Serial Peripheral Interface) Schnittstelle konfiguriert werden. Dafür gibt es bereits eine fertige Funktion. Deshalb brauchen wir uns in der Laborübung nicht weiter um die Konfiguration zu kümmern.

2.2.3 Datenübertragung zwischen DSP und Codec

Der gesamte Signallaufweg zwischen den analogen Ein- und Ausgängen und dem DSP ist in der folgenden Abbildung dargestellt. Auf dem EVB dienen zunächst analoge Operationsverstärkerschaltungen (OPV) zur Vorfilterung und Pegelanpassung der analogen Audiokanäle. Im Codec werden die analogen Signale dann mittels Sigma-Delta-Umsetzung mit sehr hoher Überabtastung digitalisiert. Eine nachfolgende Dezimationsstufe im Codec verringert die Abtastfrequenz und filtert gleichzeitig das durch die Digitalisierung entstandene Quantisierungsrauschen. Die Audiodaten der vier digitalisierten Eingangskanäle werden dann als 32-Bit-Werte über einen als I²S-Schnittstelle konfigurierten SPORT0-Kanal vom DSP eingelesen. Dabei werden, wie in der Abbildung zur Datenübertragung zu sehen, nur vier von acht möglichen Worten des Datenrahmens für die Audiodaten genutzt. Die mit „AUX_ADC …“ gekennzeichneten Worte des Datenrahmens könnten für externe AD-Wandler genutzt werden. Wir benutzen in der Laborübung jedoch nur die mit „INTERNAL ADC …“ gekennzeichneten ADCs des Audio Codecs auf dem Evaluationsboard. Vom SPORT0 werden die Daten via DMA in einen DMA-Lesepuffer (iDMARxBuffer) im DSP-Speicher geschrieben. Ist ein vollständiger Rahmen mit acht Datenworten eingelesen, wird vom SPORT0 ein Interrupt ausgelöst. Da pro Abtasttakt acht Datenworte übertragen werden, erfolgt der Interrupt genau einmal pro Abtasttakt. Sollen die Audiodaten durch den DSP verarbeitet werden, muss eine Interrupt-Service-Routine (ISR) dafür sorgen, dass die Daten in einen anderen Speicherbereich kopiert und nicht durch neue Daten überschrieben werden.

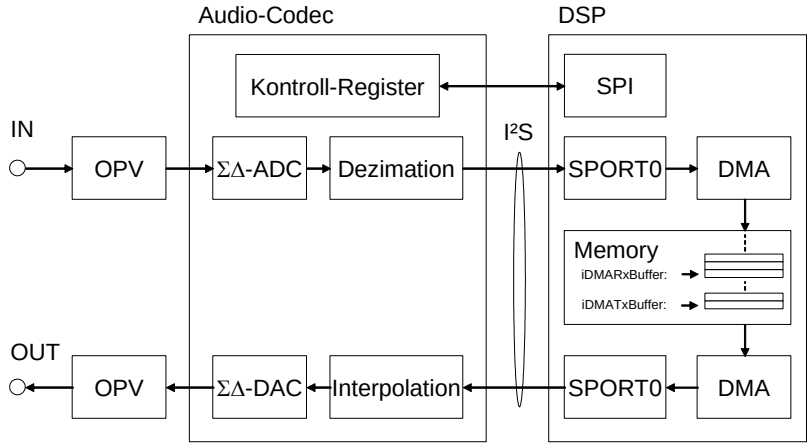

Abbildung 5: Datenaustausch zwischen Audio Codec und DSP [4]

Gleichzeitig können vom DSP verarbeitete Daten von der ISR in einen DMA-Schreibpuffer (iDMATxBuffer) im DSP-Speicher kopiert werden. Diese werden dann per DMA zu einem zweiten SPORT0-Kanal übertragen, der ebenfalls als I²S-Schnittstelle konfiguriert ist. Über diese Schnittstelle werden die Audiodaten (DAC-Data) von sechs Audiokanälen zum Audio-Codec geschickt. Im Co-

dec erfolgt zunächst eine Interpolation auf eine höhere Abtastfrequenz, bevor dann die Audiodaten per Sigma-Delta-Umsetzung in analoge Signale gewandelt werden und wiederum über Operationsverstärkerschaltungen zu den Ausgangsbuchsen des EVBs gelangen.

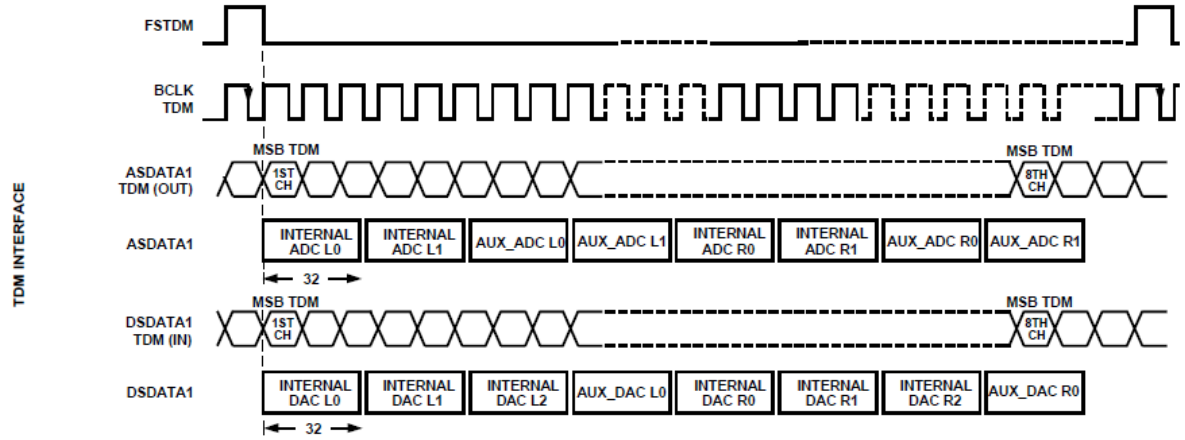

Abbildung 6: Serielle Datenübertragung zwischen DSP und Audio Codec. Die mit „AUX_“ bezeichneten externen ADCs/DACs werden im Labor nicht benutzt [8].

2.2.4 Analoge Anschlüsse des Codecs auf dem Evaluationsboard

Die folgende Abbildung zeigt die Anschlussbelegung der 4 Eingangskanäle und der 6 Ausgangskanäle des Audio Codecs auf dem Evaluationsboard. Die am in der Laborübung am häufigsten verwendeten Anschlüsse sind rot markiert.

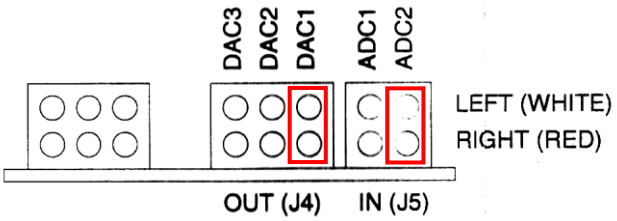

Abbildung 7: Belegung der Cinch-Stecker des Audiocodecs [5].

2.2.5 iDMARxBuffer

Hier werden die digitalen Ausgangsdaten der ADCs des Audiocodecs im internen Speicher des DSPs abgelegt. Das folgende Bild zeigt die Offsets der Speicherstellen der einzelnen ADCs. Achtung: Für die ADCs gibt es zwei Nummerierungen: eine interne und eine externe. Beispielsweise entspricht der interne ADC L0 dem externen ADC L1.

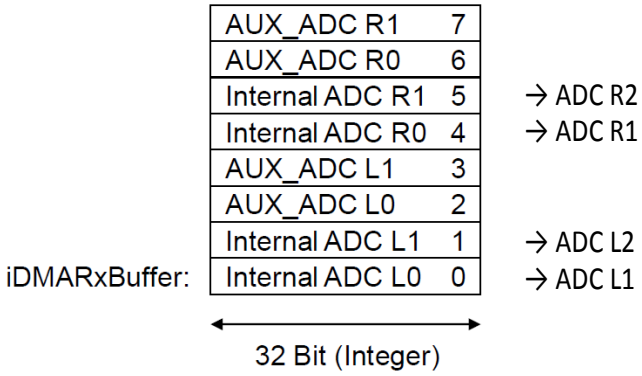

Abbildung 8: iDMARxBuffer [4].

2.2.6 iDMATxBuffer

Von hier werden die digitalen Eingangsdaten der DACs des Audiocodecs vom internen Speicher des DSPs abgeholt. Das folgende Bild zeigt die Offsets der Speicherstellen der einzelnen ADCs. Achtung: Auch hier gibt es eine interne und externe Nummerierung.

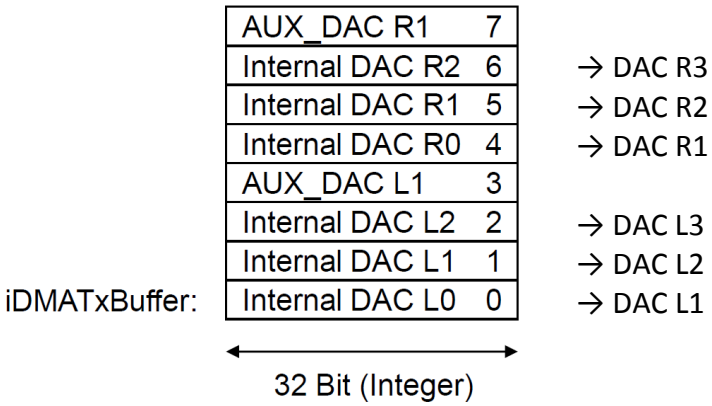

Abbildung 9: iDMATxBuffer [4].

3 Der ADSP-BF561

Der DSP „ADSP-BF561“ von Analog Devices wurde für Multimedia-Anwendungen entwickelt. Der 16-Bit Festkomma DSP vereint in einem Chip zwei DSP-Kerne (sogenannte „Blackfin Cores“), interne Speicher und eine Reihe von Schnittstellen, wie in der folgenden Abbildung dargestellt ist.

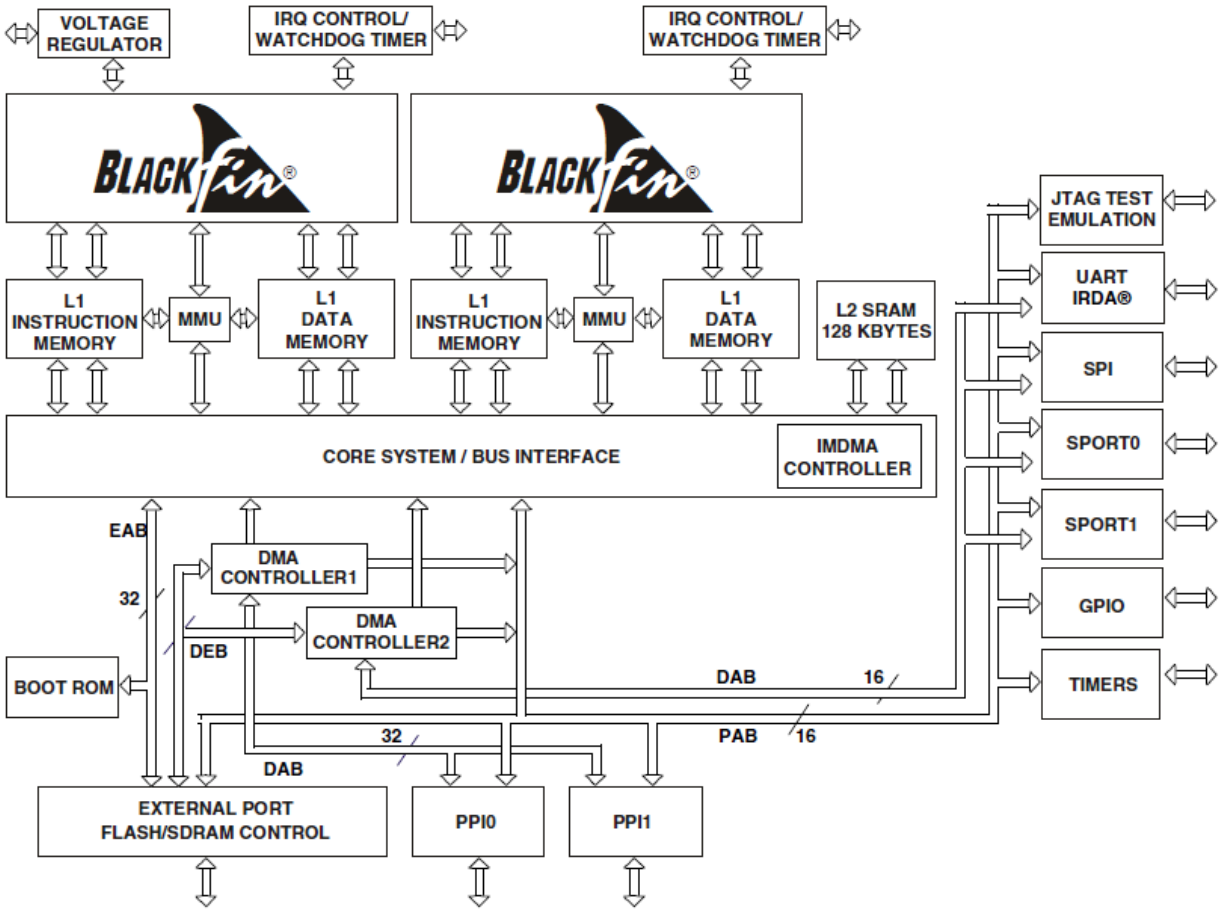

Abbildung 10: Block Diagramm des ADSP-BF561 [6].

Der DSP verfügt u.a. über die folgenden Schnittstellen:

- Parallel Peripheral Interfaces (PPIs): z.B. zum Anschluss paralleler AD/DA-Wandler oder Video Codecs.
- Ein Serial Peripheral Interface (SPI) Port: Auf dem EVB wird der SPI-Port verwendet, um den Audio Codec zu konfigurieren.
- Serial Ports (SPORTs): Serielle Schnittstellen zur Datenübertragung. Auf dem EVB erfolgt der Datenaustausch zwischen dem Audio Codec und dem DSP mit Hilfe des SPORT0.
- Universal Asynchronous Receiver Transmitter (UART): eine weitere serielle Schnittstelle.
- General Purpose I/O Pins (GPIOs, „Programmable Flags“): Sie sind auf dem EVB mit den Tastern und den LEDs verbunden.

3.1 DSP Kern – Übersicht

Die beiden identischen „Blackfin Cores“ (DSP Kerne) enthalten jeweils:

- Eine Datenrecheneinheit (Data Arithmetic Unit) mit
  - zwei 16-Bit Multiplizierern
  - zwei 40-Bit Akkumulatoren: zur Speicherung von Zwischenergebnissen
  - zwei 40-Bit ALUs: z.B. für Addition und Subtraktion
  - einem 40-Bit Barrel Shifter: für Schieboperationen
  - vier 8-Bit Video ALUs: für die effiziente Verarbeitung von Videosignalen
  - Datenregistern R0 bis R7: zum Speichern von Operanden
- Einen Adressrechner (Address Arithmetic Unit) mit
  - zwei Recheneinheiten zum Erzeugen der Datenadressen (DAG0 und DAG1, DAG=Data Address Generator)
  - Zeigerregistern (P0 bis P5, FP, SP): zum (Zwischen-) Speichern von Adressen
  - Adressregister (I0, L0, B0, M0 bis I0, L0, B0, M0): z.B. für die zyklische Adressierung

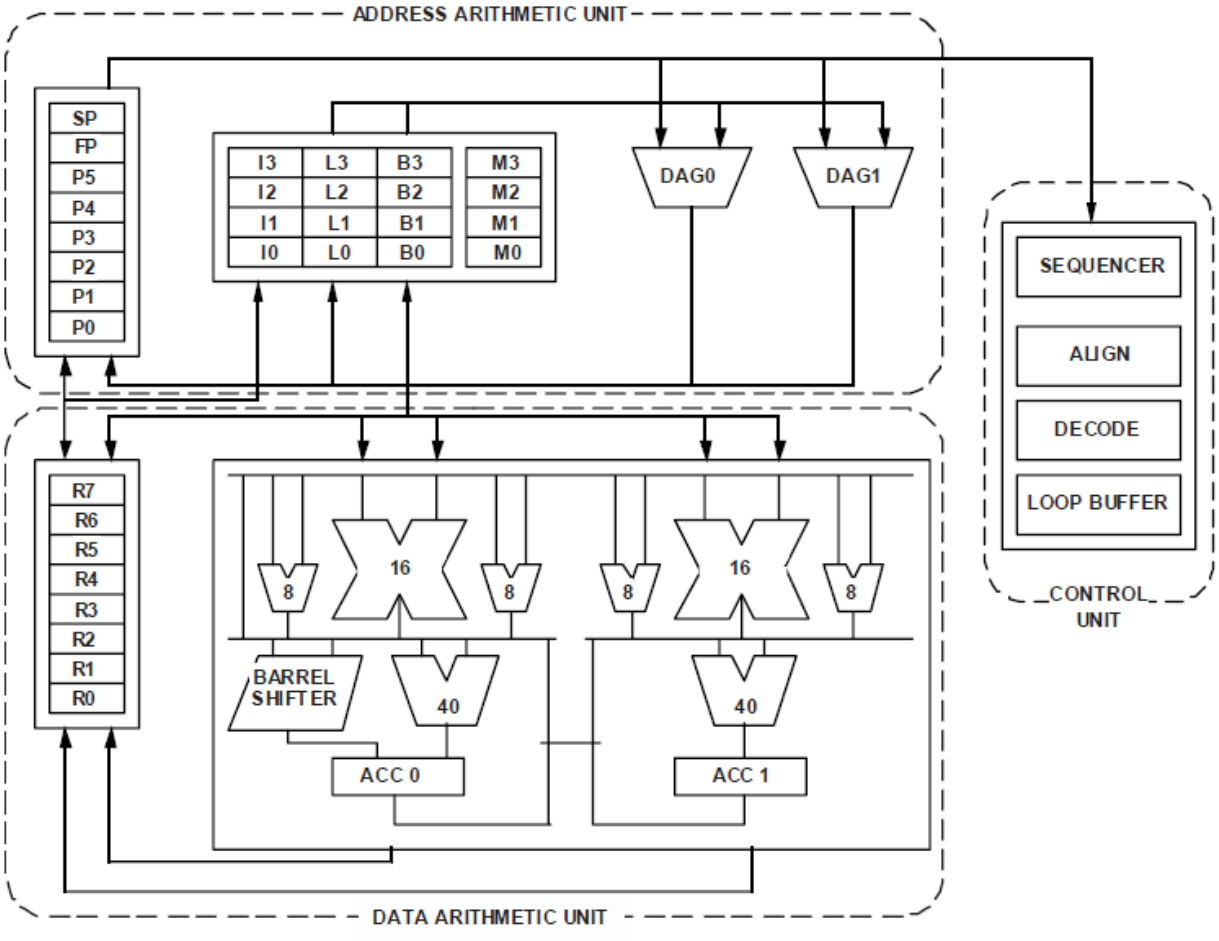

Abbildung 11: Block Diagramm eines Blackfin Cores [6].

3.2 Speicherarchitektur

Der DSP strukturiert den Speicher in einem einzigen 4 GB Adressraum mit 32 Bit Adressen. Alle Resourcen, wie z.B. interne Speicher, externe Speicher und Steuerregister, belegen getrennte Bereiche dieses gemeinsamen Adressraums. Da der DSP eine modifizierte Harvard Architektur besitzt, wird der Speicher in unterschiedliche Hierarchieebenen unterteilt. Die Level 1 (L1) Speicher arbeiten mit der vollen Taktrate des DSPs und verursachen keine (oder nur eine sehr geringe) Verzögerung. In der L1-Ebene ist der Speicher in Befehls- und Datenspeicher unterteilt. Der Befehlsspeicher enthält nur Programmanweisungen, der Datenspeicher nur Daten.

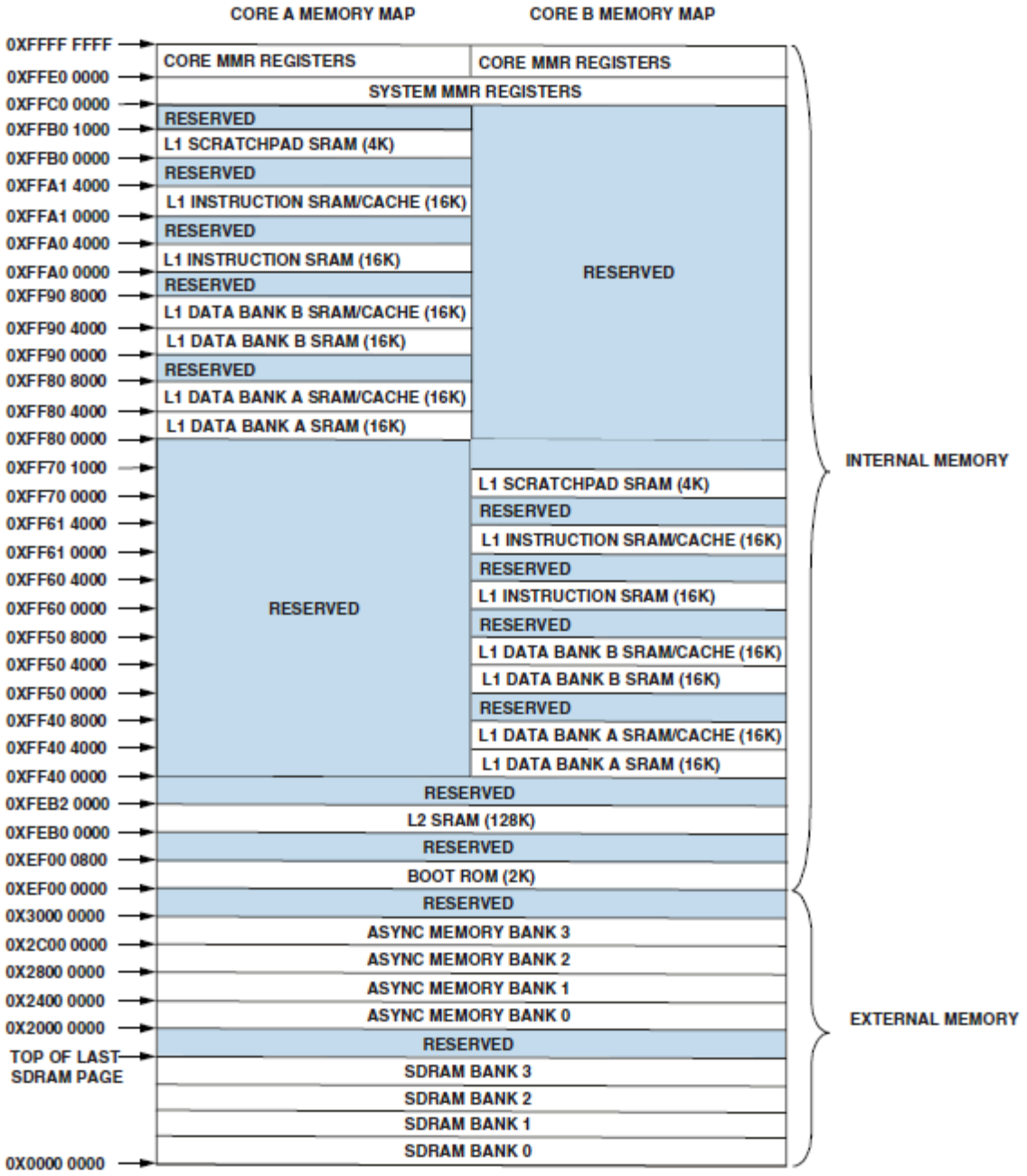

Abbildung 12: Memory-Map des DSP [6].

Die beiden Kerne des DSPs teilen sich einen gemeinsamen Level 2 (L2) Speicher, der in den Prozessorchip integriert ist. Der Zugriff auf den L2 Speicher verursacht größere Verzögerungen als der Zugriff auf den L1 Speicher. Der L2 Speicher ist deutlich billiger und es steht eine größere Speicherkapazität auf dem Chip zur Verfügung. Beim L2 Speicher handelt es sich um einen gemeinsamen Befehls- und Datenspeicher, d.h. er kann sowohl Programmanweisungen als auch Daten in beliebiger Mischung speichern.

3.3 Recheneinheiten (Computational Units)

Im Folgenden werden die wichtigsten Recheneinheiten des DSPs sowie Beispiele für Befehle, welche mit diesen Recheneinheiten ausgeführt werden können, besprochen. Der DSP ist für 16-Bit Festkommaoperationen optimiert. Negative Zahlen werden als Zweierkomplement dargestellt. Das häufigste Zahlenformat ist das 1.15-Format. Die meisten der einfachen (häufig vorkommenden) Befehle werden als 16-Bit Wörter codiert. Es gibt aber auch kompliziertere Befehle, die mit 32-Bit Wörtern codiert werden.

3.3.1 Register

Die folgende Abbildung zeigt eine Übersicht über die Register des DSPs.

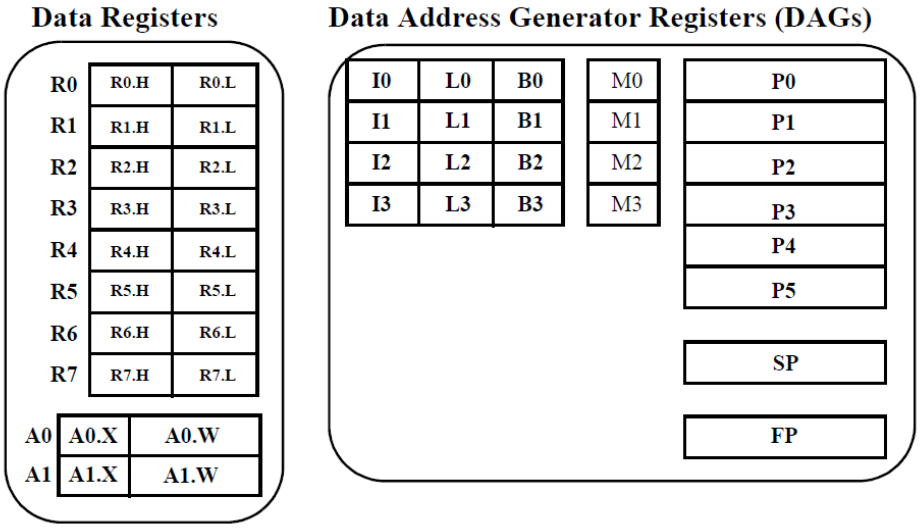

Abbildung 13: Register des Blackfin Cores [6].

Datenregister (Data register file)

Der DSP verfügt über 8 Datenregister R0 bis R7. Sie dienen vorwiegend zum Speichern der Eingangsoperanden und der Ergebnisse von Rechenbefehlen. Jedes dieser Register kann ein 32-Bit Wort speichern. Die Register können aber auch in 16 unabhängige 16-Bit Register geteilt werden. Die höherwertigen 16 Bits werden dann mit Rn.H bezeichnet, die niederwertigen 16 Bits mit Rn.L, z.B. R1.H und R1.L.

Akkumulatoren (Accumulator Registers)

Der DSP verfügt über zwei Akkumulatoren A0 und A1. Sie dienen vorwiegend zum Speichern der Ergebnisse von Rechenoperationen, die in den ALUs ausgeführt wurden. Jeder dieser Akkumulatoren kann ein 40-Bit Wort speichern. Am Beispiel von A0 wird im Folgenden gezeigt, wie die 40 Bits der Akkumulatoren in kleinere Einheiten geteilt werden können:

- A0.X steht für die höherwertigen 8 Bits (e**X**tension bits).
- A0.W steht für die niederwertigen 32 Bits. (das niederwertige 32-Bit **W**ort)
- A0.H steht für die höherwertigen 16 Bits von A0.W.
- A0.L steht für die niederwertigen 16 Bits von A0.W.

Zeigerregister (Pointer register file)

Der DSP verfügt über sechs Zeigerregister (P-registers) P0 bis P5. Sie haben eine Breite von 32 Bits und dienen vorwiegen zur Speicherung von Adressen. Zum „pointer register file“ gehört außerdem der „Stack Pointer“ SP und der „Frame Pointer“ FP.

DAG Register (Data Address Generation Registers)

Die DAG Register dienen in erster Linie zur Adressierung von Daten. Die unterschiedlichen Register werden benutzt, um

- Indexadressen zu speichern: I[3:0].
- Modifikationswerte zu speichern: M[3:0].
- Basisadressen von Ringpuffern zu speichern: B[3:0].
- die Länge von Ringpuffern zu speichern: L[3:0].

Eine der wichtigsten Anwendungen der DAG Register ist die Implementierung von Ringpuffern (cyclic buffers) für die Realisierung von FIR Filtern. Das wird im Versuch 1 näher besprochen.

3.3.2 Arithmetic Logic Unit (ALU)

Jeder der beiden DSP Kerne verfügt über zwei ALUs, ALU0 und ALU1. Für die meisten Befehle wird die ALU0 eingesetzt. Die ALU1 kommt nur bei parallelen Operationen zum Einsatz.
Die ALUs werden für die folgenden Operationen eingesetzt:

- Festkomma Addition und Subtraktion von Registern und Konstanten („immediate values“).
- Akkumulation und Subtraktion von Produkten aus den Multiplizierern.
- Logische Verknüpfungen wie: AND, OR, NOT, XOR, bitweises XOR
- Funktionen, wie ABS, MAX, MIN und Round

Eine Übersicht über mögliche Eingangsoperanden und Ausgangsoperanden für eine ALU gibt die folgende Tabelle.

Input	Output
Two or four 16-bit operands	One or two 16-bit results
Two 32-bit operands	One 32-bit result
32-bit result from the multiplier	Combination of 32-bit result from the multiplier with a 40-bit accumulation result

Abbildung 14: Befehlsklassen [6].

Es gibt folgende Klassen von Befehlen, die teilweise beide ALUs nutzen:

1. Einfache 16-Bit Operationen: verknüpfen zwei 16-Bit Eingangsoperanden zu einem 16-Bit Ergebnis, Beispiel: R3.H = R1.H + R2.L (NS); – Addition ohne Sättigung (NS: no saturation)
2. Doppelte 16-Bit Operationen: Eingangsparameter sind hier zwei 32-Bit Datenregister, Ausgangsoperand ist ein 32-Bit Datenregister. Die Eingangsoperanden werden jedoch als vier unabhängige 16-Bit Eingangswerte und der Ausgangsoperand als zwei unabhängige 16-Bit Ausgangswerte betrachtet:
   Beispiel: R3 = R1 +|- R2 (S); entspricht R3.H = R1.H +R2.H (S); R3L = R1L – R2L (S); mit Sättig.
3. Vierfache 16-Bit Operationen, Beispiel: R3 = R0 +|+ R1, R2 = R0 –|– R1 (S) ; entspricht den folgenden 4 Operationen

```
R3.H = R0.H + R1.H (S);
R3.L = R0.L + R1.L (S);
R2.H = R0.H – R1.H (S);
R2.L = R0.L – R1.L (S);
```

4. Einfache 32-Bit Operationen: Beispiel: R3 = R1 + R2 (NS); Addition ohne Sättigung
5. Doppelte 32-Bit Operationen: Beispiel: R3 = R1 + R2, R4 = R1 – R2 (NS); Addition und Subtraktion von R1 und R2 in einem Befehl.

3.3.3 Barrel Shifter

Führt u.a. folgende Operationen durch

- Arithmetische Shifts
- Logische Shifts
- Bit Rotationen

Unterschied zwischen arithmetischen und logischen Shifts: Bei einem arithmetischen Rechtsshift werden auf der linken Seite Kopien des Vorzeichenbits eingeschoben, beim logischen Rechtsshift werden dagegen auf der linken Seite Nullen eingeschoben.

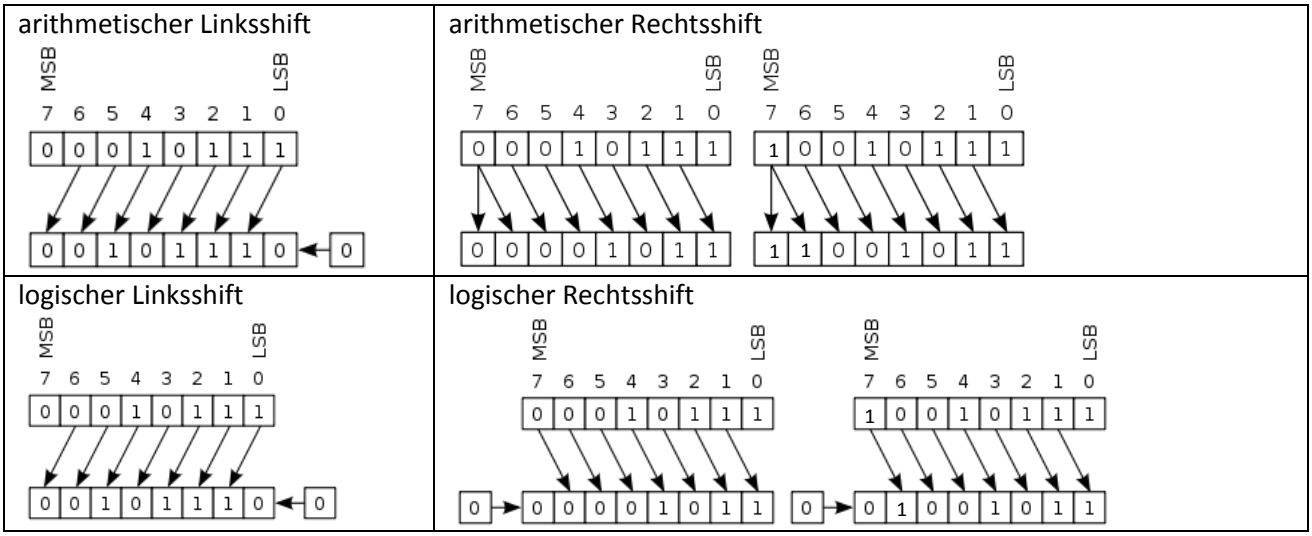

Abbildung 15: Veranschaulichung der Shift-Befehle [9].

3.4 Programmable Flags

Der DSP verfügt über 3·16=48 Ein-/Ausgabepins, sogenannte „Programmable Flags“ (PFs). Jeder dieser Pins kann einzeln als Eingang oder Ausgang konfiguriert werden. Das Verhalten der PFs wird durch die folgenden Registersätze gesteuert. Für die 48 PFs gibt es von jedem der nachfolgend besprochenen Registersätze drei 16-Bit Einzelregister: FIO0_..., FIO1_..., FIO2_... . Die folgenden Abbildungen zeigen jeweils nur eine Auswahl der Einzelregister.

- **Flag Direction Register (FIOn_Dir)**: Jedes Bit des Registers ist für ein PF zuständig. Eine logische 0 konfiguriert den entsprechenden Pin als Eingang, eine logische 1 als Ausgang. In den folgenden Abbildungen ist auch eingezeichnet, an welchen Pins des Evaluationsboards die Taster und die LEDs angeschlossen sind.

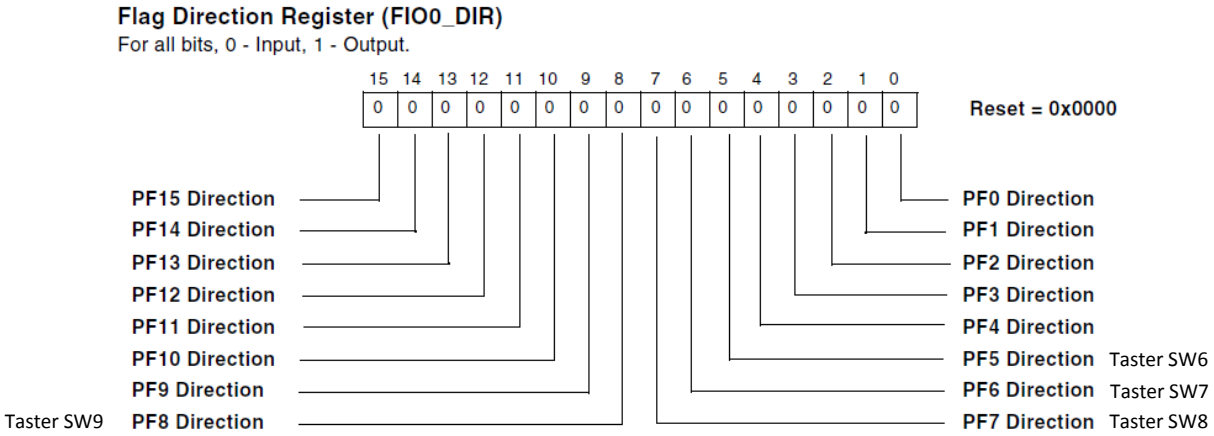

Abbildung 16: Flag Direction Register FIO0_DIR [6].

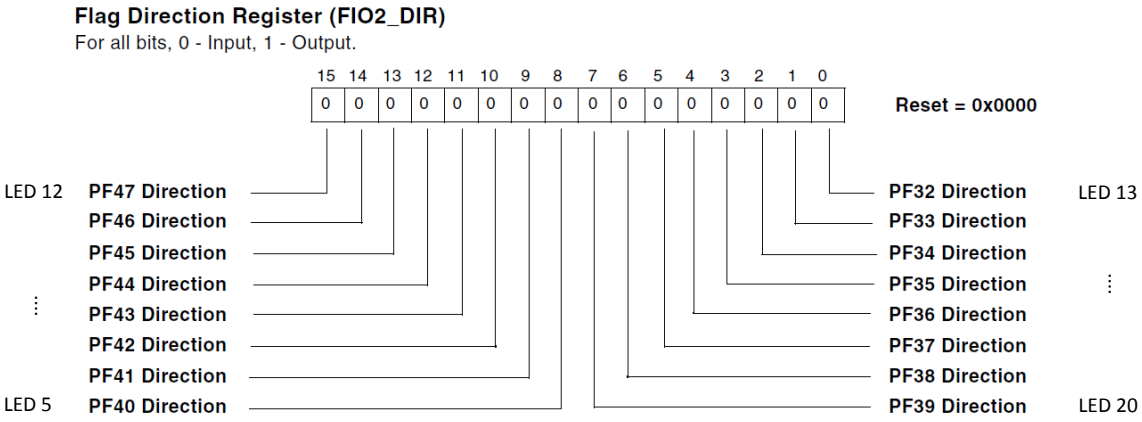

Abbildung 17: Flag Direction Register FIO2_DIR [6].

- **Flag Input Enable Register (FIOn_INEN):** schaltet den „Input Buffer“ für Eingangspins ein. Standardmäßig sind alle „Input Buffers“ ausgeschaltet.

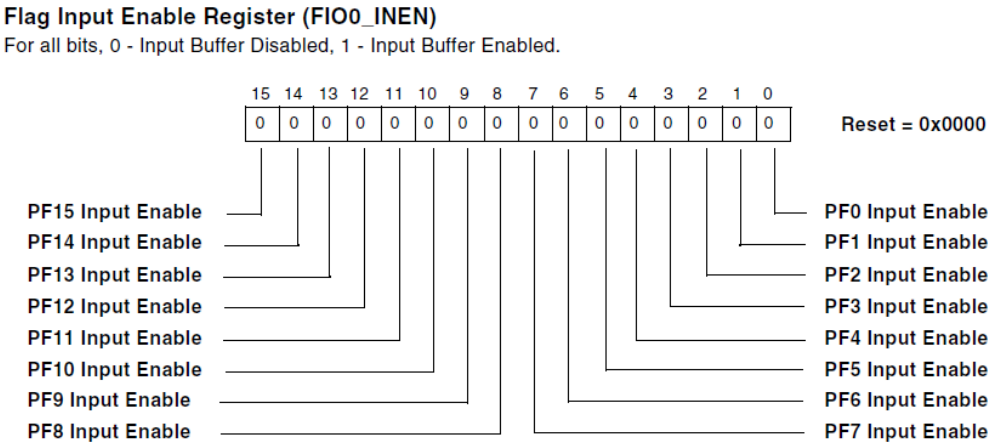

Abbildung 18: Flag Input Enable Register FIO0_INEN [6].

- Flag Data Register (FIOn_FLAG_D): Für Ausgabepins wird durch Beschreiben des Flag Data Registers der Zustand des Pins festgelegt. Für Ein-/und Ausgabepins kann der Zustand des Pins aus dem Flag Data Register ausgelesen werden.

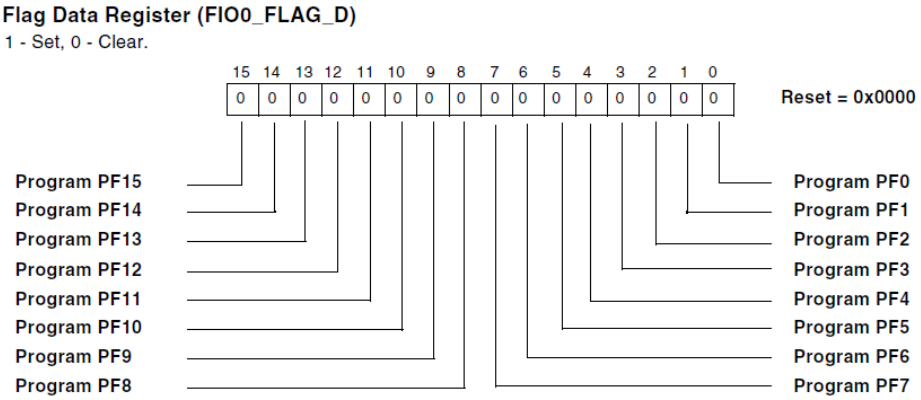

Abbildung 19: Flag Data Register FIO0_D [6].

4 Literatur

[1] G. Doblinger: Signalprozessoren. J. Schlembach Verlag, 2004

[2] R. Dörfel, D. Menz: Signalprozessoren: Skript des Instituts für Informatik VI, Technische Universität München

[3] Analog Devices: ADSP-BF561 EZ-KIT Lite Evaluation System Manual, Revision 3.3, 2012

[4] M. Purat: Vorbereitende Aufgaben zum DSP Labor, Beuth Hochschule für Technik Berlin, 2009

[5] M. Purat: DSP-Labor – Durchführung und Auswertung, Beuth Hochschule für Technik Berlin, 2008

[6] Analog Devices: ADSP-BF561 Blackfin® Processor Hardware Reference, Revision 1.0, 2005

[7] Analog Devices: ADSP-BF53x-BF56x Blackfin® Processor Programming Reference, Revision 1.1, 2006

[8] Analog Devices: Datenblatt des Audio-Codecs AD1836A

[9] wikipedia, 2013