



Rekonfigurierbare Rechnerplattform

„Car Infotainment“

Dr. Karlheinz Weiß

Inhaltsverzeichnis

1	Einführung	3
2	Vor- und Nachteile von ASIC bzw. FPGA-Implementierungen	3
2.1	ASIC-Implementierung	3
2.2	FPGA-Implementierung	3
2.3	Fazit	4
3	FPGA-Architekturen	4
3.1	Implementierungsalternativen	4
3.1.1	Compile Time Rekonfiguration (CTR)	4
3.1.2	Globale Run Time Rekonfiguration (gRTR)	4
3.1.2.1	Partiale Run Time Konfiguration (pRTR)	5
4	Vor und Nachteile von Entwurfsmethoden	5
4.1.1	Sequentielle Entwurfsmethode	5
4.1.2	Parallele Entwurfsmethode	5
4.1.2.1	Toolset SPYDER	6
5	Rekonfigurierbare Car Infotainment Plattform	6
5.1	Gesamtarchitektur	7
5.2	Anwendung der parallelen Entwurfsmethodik	8
5.2.1	MOST parallel combined Mode	8
5.2.2	3-D Grahik Modul	9
6	Fertiges Zielsystem	10
7	Zusammenfassung	10
8	Ausblick	10

Abbildungen

Abbildung 1:	Sequentieller Entwicklungsablauf	5
Abbildung 2:	Paralleler Entwicklungsablauf	6
Abbildung 3:	Grundstruktur eines eingebetteten Systems	6
Abbildung 4:	Blockdiagramm BASE BOARD	7
Abbildung 5:	Blockdiagramm MAINBOARD	7
Abbildung 6:	Blockdiagramm FPGA	8
Abbildung 7:	MOST Blockdiagramm	8
Abbildung 8:	Entwicklungsaufbau MOST auf SPYDER-CORE	9
Abbildung 9:	Blockdiagramm 3-D-Grahikmodul	9
Abbildung 10:	Entwicklungsaufbau 3-D Graphik auf SPYDER-CORE	9
Abbildung 11:	CAR Infotainment Plattform Version 1.	10

1 Einführung

Die neuesten Modelle führender Automobilhersteller der Oberklasse bieten Ihren Kunden sogenannte „Car-Infotainment“ Systeme an. Darunter versteht man die Verschmelzung von klassischen Audio-Geräten wie CD-Spieler, Radio- und Verstärker mit neueren Geräten im Fahrzeug wie Navigationssystem, MP3-Player, Telefon und darauf basierendem Internet-Zugang. Alle Geräte werden über ein gemeinsames Mensch-Maschinen-Interface (MMI) bedient. Dabei können physikalisch alle Geräte in einem Gehäuse untergebracht sein oder verteilt auf mehrere Gehäuse, die dann über entsprechende Bussysteme miteinander vernetzt werden. Als gemeinsamer Standard aller führenden deutschen Hersteller hat sich der optoelektronische MOST-Bus (Media Oriented System Transport) als Kommunikations-Bus etabliert. Zusätzlich dient der CAN-Bus (Controller Area Network) als Verbindung zur klassischen Kraftfahrzeugelektronik. Die Verbindung beider Bus-Systeme sowie der Zugriff auf Internet-Dienste über Mobiltelefon-Modul ermöglichen neue, innovative Telematik-Dienste

Kernstück eines solchen Infotainment-Systems ist eine entsprechend leistungsstarke Rechnerplattform, die über entsprechende Schnittstellen Zugriff auf viele periphere Hardware-Komponenten hat. Um die Integration möglichst hoch und die Kosten niedrig zu halten, werden dafür anwendungsspezifische Schaltungen benötigt. Diese können entweder in einem „Field Programmable Gate Array (FPGA)“ oder auch in einem „Application Specific Integrated Circuit (ASIC)“ implementiert werden.

In diesem Artikel werden zunächst in Kapitel 2 die Vor- und Nachteile der beiden Technologien dargelegt und erklärt, warum ein „rekonfigurierbarer“ FPGA-Lösungsansatz gewählt wurde. In Kapitel 3 werden verschiedene FPGA-Technologien und damit verbunden Implementierungsalternativen für rekonfigurierbare Systeme dargestellt. Im Kapitel 4 werden zwei verschiedene Entwicklungsmethoden (sequentiell und parallel) dargestellt und verglichen. Dabei wird erläutert, warum die sogenannte parallele Entwicklungsmethode bei komplexen eingebetteten Systemen wesentliche Vorteile bietet. Kapitel 5 zeigt die Gesamtarchitektur der rekonfigurierbaren CAR Infotainment Plattform“ und erläutert an zwei ausgewählten Beispielen die Anwendung der in Kapitel 4 vorgestellten parallelen Entwurfsmethodik. In Kapitel 6 werden die wichtigsten Aussagen nochmals zusammengefasst und Kapitel 7 gibt einen Ausblick auf zukünftig mögliche Entwicklungen, welche die vorgestellte Architektur und verwendete Technologie bietet.

2 Vor- und Nachteile von ASIC bzw. FPGA-Implementierungen

Eine Car Infotainment Plattform zeichnet sich durch die Integration vieler verschiedener Module aus. Daraus folgt die notwendige Implementierung vieler verschiedener Schnittstellen. Um die Kosten weiter zu reduzieren und die Integrationsdichte zu erhöhen ist es vorteilhaft, diese Schnittstellen direkt mit auf einem entsprechenden Embedded Controller-Chip zu integrieren. Prinzipiell wären dazu zwei technologische Weg möglich, entweder die Implementierung von Prozessor und Schnittstellen in einem einzigen ASIC-Chip oder in einem FPGA der neuesten Generation mit eingebettetem Microcontroller-Kern. Dieses Kapitel beleuchtet die Vor- und Nachteile beider Lösungswege.

2.1 ASIC-Implementierung

Die durch die Technologie bedingten Vorteile von ASIC's liegen in den geringeren Kosten, sobald eine bestimmte Stückzahl überschritten wird. Zusätzlich hat man durch eine hohe Integrationsdichte in einem Chip Vorteile im Bereich elektromagnetischer Verträglichkeit (EMV), geringeren Flächenbedarf auf der Leiterplatte sowie kleineren Leistungsbedarf.

Nachteilig wirken sich die größeren Entwicklungskosten aus, was allerdings immer relativ zu den zu erwartenden Stückzahlen zu sehen ist. Als entscheidender Nachteil hat sich in der Vergangenheit das drastisch steigende Entwicklungsrisiko gezeigt. Auf Grund der kurzen Entwurfszeiträumen können Entwicklungsfehler, die bei der Prototypenproduktion nicht bzw. erst spät im Entwicklungsprozeß erkannt werden, ganze Entwicklungsprojekte verzögern bzw. unter extremen Bedingungen sogar scheitern lassen. Entwicklungsfehler können auf Grund der „festen“ Technologie nur mit größtem Aufwand, wie z.B. einem teureren und zeitaufwendigem Re-Design-Schritt, behoben werden. Zur Zeit leiden viele Automobilhersteller unter dem Umstand, dass ihre Zulieferer diese Entwicklungsprobleme unterschätzt haben und nur langsam in den Griff bekommen. Als gravierend ist dabei die Tatsache, das ohne eine fehlerfreie Hardware eine nur eingeschränkte Software-Entwicklung auf dem Zielsystem beginnen kann, was wesentlich zum Zeitverzug beiträgt.

Wenn der im wesentlichen durch die kommerzielle PC-Welt getriebene Entertainment-Sektor mit dem klassischen Audio-Sektor der Automobilwelt zusammentrifft, entstehen ständig neue Forderungen an spezielle Schnittstellen zur Integration neuer Module und Funktionalitäten. Ein nach der Produktion festgelegter ASIC-Baustein kann sich diesen Entwicklungen nur schwer anpassen, was eine gewisse Einschränkung in der Flexibilität der Systemarchitektur zur Folge hat. Dadurch würde für jedes Nachfolge-Projekt ein neuer Maskengang notwendig, um neue oder geänderte Schnittstellen zu integrieren. Diese Tatsache bewirkt aber gleichzeitig eine Stückzahl-Reduzierung pro Chip, wenn der ASIC nicht ohne Änderung für mehrer verschiedene Infotainment-Systeme verwendet werden kann, was die Kosten für einen ASIC-Chip erhöht.

2.2 FPGA-Implementierung

Der wesentliche Vorteil einer FGPA-Lösung sind die schnelleren Entwicklungszyklen (Time to Market) und geringeren Entwicklungskosten. Entwicklungsfehler in der digitalen Schaltung können mit geringem Aufwand auch noch in späteren Entwicklungsstadien behoben werden, was das Entwicklungsrisiko wesentlich senkt. Sogar nach Auslieferung des Systems an den Kunden können Hardware-Fehler im Rahmen üblicher Software-Updates noch

behalten werden. Ferner bieten neueste FPGA-Generationen bereits die Möglichkeit, Prozessor und digitale Peripherie in einem Chip zu integrieren. Neue Schnittstellen können noch während eines laufenden Projekts bzw. im Nachfolgeprojekt mit geringem Aufwand berücksichtigt werden, was zu signifikanten Freiheitsgraden für die Systemarchitektur führt.

Um eine entsprechende Logik-Funktion programmierbar zu realisieren sind wesentlich mehr Silizium-Ressourcen notwendig wie in einem ASIC. Diese führt im Verhältnis zu einer wesentlich größeren Chip-Fläche. Daraus resultiert eine größere Stromaufnahme und auch höherer Stückkosten. Die FPGA-Kosten steigen zusätzlich durch die Programmierkosten (bei einer sogenannten Anti-Fuse-Technik) bzw. durch externe Flashspeicher bei SRAM-basierten FPGAs (bei modernen FPGAs mehrere 100kByte) zur Speicherung der Konfigurationsdaten. Ferner ist eine entsprechende Firmware notwendig, um das System beim Einschalten der Betriebsspannung zum Leben zu erwecken bzw. auch im Rahmen der erwähnten Software-Updates auch ein Update der Konfigurationsdaten durchzuführen.

2.3 Fazit

Welcher technologische Ansatz der Bessere ist, kann allgemein nicht festgelegt werden. In der hier beschriebenen Arbeit wird versucht, die Vorteile beider Varianten zu nutzen. Da das Gebiet des „Car Infotainments“ noch relativ neu und in der Entwicklung ist, wird zum momentanen Zeitpunkt ein FPGA-basierter Ansatz gewählt. Durch die Möglichkeit, Prozessor und Peripherie-Schaltungen auf einem FPGA-Chip zu integrieren und die Auswahl von modernen Entwicklungswerkzeugen (Schaltungssynthese aus VHDL oder Verilog) können nach Markteinführung und Marktakzeptanz durch begrenzten Aufwand auch ASIC-basierte Lösungen implementiert werden. In letzter Konsequenz wird dann abhängig von einer bestimmten Stückzahl die kostengünstigste Lösung sich durchsetzen, die heute noch nicht vorhergesagt werden kann. Wichtig ist, dass durch Auswahl von entsprechenden FPGA-Architekturen und dazu gehörende Entwicklungsmethoden ein entsprechender Transfer von FPGA nach ASIC mit möglichst kleinem Änderungsrisiko durchgeführt werden kann. Der folgende Text zeigt diese Strategie anhand eines entsprechend komplexen Beispiels einer „Car Infotainment Plattform“.

3 FPGA-Architekturen

Prinzipiell unterscheidet man zwischen einmalig programmierbaren und unbegrenzt wiederprogrammierbaren Bausteinen. Die einmalig programmierbaren FPGAs basieren auf vielen kleinen Sicherungen (sogenannte Anti-Fuse-Technologie [Act02]), welche durch einen gezielten Stromstoß durchgebrannt werden und somit eine bestimmte Logik-Verknüpfung bzw. auch elektrische Leitungsverbindung zur Verdrahtung eines Chips realisieren. Diese Bausteine müssen vor der Leiterplatten-Produktion programmiert werden. Sie werden in dieser Arbeit nicht verwendet und sollen hier nur der Vollständigkeit erwähnt werden

Bei wiederprogrammierbaren Bausteinen dienen statische SRAM-Zellen (wie in Speicher-Chips) zur Speicherung von Bits, die angelegt an entsprechende digitale Schaltungen logische Verknüpfungen realisieren (sogenannte Complex Logic Blocks - CLBs). Die Verdrahtung wird ebenfalls mit gespeicherten Bits realisiert, in dem sie an entsprechende Schalttransistoren angelegt werden, die dann Leitungssegmente verbinden bzw. trennen. Diese Tatsache bedeutet, dass nach Einschalten dieser Bausteine völlig funktionslos sind und zunächst mit einem Datensatz geladen (konfiguriert) werden müssen, bevor sie eine entsprechende Hardware-Funktion ausführen können. Tiefere Informationen können in [Xil98] und [Alt02] gefunden werden.

In den letzten 10 Jahren hat die FPGA-Technologie eine stürmische Entwicklung erlebt. Am Anfang waren Bausteine mit max. 10.000 Gattern-Aquivalenten verfügbar. Heute sprechen die Hersteller von bis zu 10 Mio. Gattern, allerdings ist die Zählweise nicht immer genau nachvollziehbar. Im Vergleich zu ASIC's muss man diese Angaben etwa durch einen Faktor 4-5 teilen, um vergleichbare Verhältnisse zu bekommen.

Die ersten FPGA's (wie die Xilinx XC4000-Familie oder Altera Flex 10k-Familie) eigneten sich nur für die Implementierung von digitaler Logik. Die Architekturen der nächsten Generation enthielten dann auch eingebettete RAM-Strukturen zur effizienten Integration von On-Chip-Speicher (z.B. Xilinx Virtex-Familie). Moderne FPGA-Architekturen der neuesten Generation wurden zum System-On-Chip (SoC) weiterentwickelt (Integration von 32 Bit Embedded Controller, z.B.: Virtex-II PRO [Xil02]).

3.1 Implementierungsalternativen

Im allgemeinen muss unterschieden werden zwischen den verschiedenen FPGA-Architekturen und den verschiedenen Konfigurationsalternativen, welche benutzt werden, um Hardware-Module in FPGA's zu implementieren. Im folgenden Text werden diese Alternativen kurz erläutert.

3.1.1 Compile Time Rekonfiguration (CTR)

Bei dieser Konfigurationsalternative wird ein erzeugter Konfigurationsdatensatz „einmalig“ in ein FGPA geladen. Bei Anti-Fuse-FPGAs muß dieser vor der Produktion der Leiterplatte programmiert werden, bei SRAM-basierten FPGAs muss er unmittelbar nach dem Einschalten der Betriebsspannung von z.B. einem Mikrocontroller aus einem externen Flash ins FPGA geladen werden und bleibt dann während der gesamten Betriebszeit bis zum Abschalten ohne Änderung erhalten. Damit ist diese Alternative für alle FPGA-Technologien anwendbar und wird zur Zeit in den meisten Projekten verwendet.

3.1.2 Globale Run Time Rekonfiguration (gRTR)

Bei dieser Konfigurationsalternative werden zur Laufzeit des Systems die Konfigurationsdaten ausgetauscht. Wenn Hardware-Module nicht gleichzeitig gebraucht werden (z.B. ein ATAPI-Schnittstelle und eine Secure Digital Card-Schnittstelle), kann ein Mikroprozessor die FPGA-Funktionalität während der Laufzeit austauschen. Der Zusatz

„global“ bedeutet, dass der gesamte FPGA-Chip in seiner Funktionalität geändert wird (im Gegensatz zu „partial“, was im nächsten Abschnitt behandelt wird).

Diese Alternative kann somit nur auf SRAM-basierenden FPGA's angewendet werden. Der Vorteil dabei ist, dass auf einer gegebenen FPGA-Chip-Fläche mehr Funktionalität abgebildet werden kann. Voraussetzung dafür ist, dass die Funktionen natürlich nicht gleichzeitig, sondern zeit-exklusiv benötigt werden. Zu beachten ist, dass ein Austausch der gesamten Konfigurationsdaten ca. 10ms dauern kann, d.h. nur in größeren Zeitintervallen Sinn macht.

3.1.2.1 Partiale Run Time Konfiguration (pRTR)

FPGA-Chips der neuesten Generation ermöglichen nicht nur den Austausch der gesamten Konfiguration, sondern ermöglichen die Re-Konfiguration von bestimmten Bereichen des FPGA-Chips, während der restliche Teil des FPGAs mit der gegebenen Konfiguration weiterläuft. Damit wären z.B. Anwendungen denkbar, bei denen nach Bedarf nur bestimmte Schnittstellen-Komponenten ausgetauscht werden, wenn auf externe Peripherie zugegriffen wird.

Damit ändert die pRTR das zeit-exklusive Konzept der gRTR und ermöglicht so den Austausch von verschiedenen Subsets der Hardware, welche jeweils eine bestimmte Chip-Fläche nur zu einer bestimmten Zeit benutzen. Dies führt zu einer noch feineren, partiell rekonfigurierbaren Systemarchitektur und einer wesentlich effektiver Nutzung des FPGA-Chips. Diese Alternative ist zur Zeit noch Gegenstand der Forschung und wird in Zukunft die vorher genannten Alternative teilweise ablösen. Tiefergehende Informationen sind in zu finden [HuWi95] und [WeKiRo98][WeKiRo98][WeKiRo98].

4 Vor und Nachteile von Entwurfsmethoden

Dieses Kapitel betrachte zwei verschiedene Entwurfsmethoden, welche für die Entwicklung von komplexen eingebetteten Systemen benutzt werden. Der Einsatz von FPGA's beeinflusst diese Methoden wesentlich. In Kapitel 5 wird diese Tatsache anhand konkreter Beispiele beleuchtet.

4.1.1 Sequentielle Entwurfsmethode

In Abbildung 1 ist der Ablauf über die Zeit der sogenannten sequentiellen Entwurfsmethode dargestellt. Nach der Spezifikation wird eine Partitionierung in Hardware und Software durchgeführt. Danach wird zuerst eine Hardware-Architektur festgelegt und implementiert. Danach wird auf dieser Hardware (auch Ziel-System genannt) die entsprechende Software-Architektur definiert und ebenfalls implementiert. Zum Schluss findet eine Integration in die Umwelt und ein Abschlusstest statt.

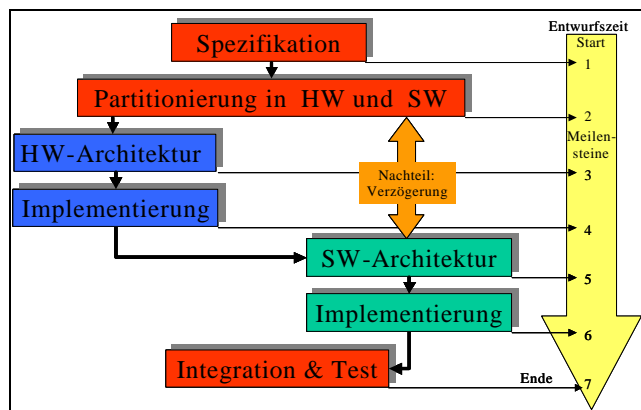


Abbildung 1:
Sequentieller Entwicklungsablauf

Zunächst kann man feststellen, dass diese Methode wohl am weitesten verbreitet ist. Die Vorteile dieser Methode liegen darin, dass man möglichst schnell zu einem Hardware-Prototypen kommt, weshalb viele Entwickler sie auch als besonders „ergebnisorientiert“ bezeichnen.

Wenn in der späteren Software-Entwicklungsphase Probleme auftauchen, die am Anfang auf Grund der Komplexität nicht erkannt wurden, ist eine Fehlerbehebung sehr aufwendig, welche in extremen Fällen sogar das Projekt aus Zeit- oder Kostengründen zum Scheitern bringen kann. Man kann sagen, dass das Entwicklungsrisiko ist unkalkulierbar ist, da erst ein sehr später Einblick in das tiefere Systemverhalten möglich ist.

Um diesem Problem zu begegnen, reagieren die Entwickler mit einer Einschränkung der Bauteilwahl. Diese Tatsache bedeutet, man ist bestrebt aus früheren Projekten nur bekannte Komponenten einzusetzen, um die Probleme zu begrenzen. Dadurch werden aber neuere, innovativere Komponenten nicht in die Systemarchitektur mit einbezogen, was zu sub-optimalen Lösungen führt.

4.1.2 Parallele Entwurfsmethode

In Abbildung 2 ist ein paralleler Entwicklungsablauf dargestellt, wie er auch für das komplexe Car-Infotainment System in Kapitel 5 benutzt wurde. Nach der Spezifikation der Funktionalität wird eine initiale Partitionierung und Komponentenauswahl getroffen. Danach findet eine zweistufige Emulation jeder Einzelkomponente statt. In der ersten Stufe wird die Komponente anhand verfügbarer Informationen bewertet (z.B. Datenblatt, User Manual), danach wird die Funktionsweise der Komponenten in einer Echtzeit-Emulation auf ihre wirkliche Eignung überprüft. Dazu wird eine entsprechende Rapid Prototyping Plattform benutzt, welche im nächsten Abschnitt beschrieben ist.

Bestätigt die Emulation die getroffenen Angaben aus dem Datenblatt, wird sie in die Systemarchitektur übernommen. Sollte sie die Anforderungen nicht einhalten oder treten signifikante Probleme auf, kann kurzfristig die Iterationsschleife mit einer alternativen Komponente durchlaufen werden. Zum Schluss findet eine Integration der

emulierten Einzelkomponenten zum Gesamtsystem statt. Entscheidender Unterschied ist, dass in diesem Ablauf die Hardware des eigentlichen Gesamtsystems zum Schluss entsteht, im Gegensatz zur sequentiellen Methode, die sofort mit der Implementierung des Zielsystems startet.

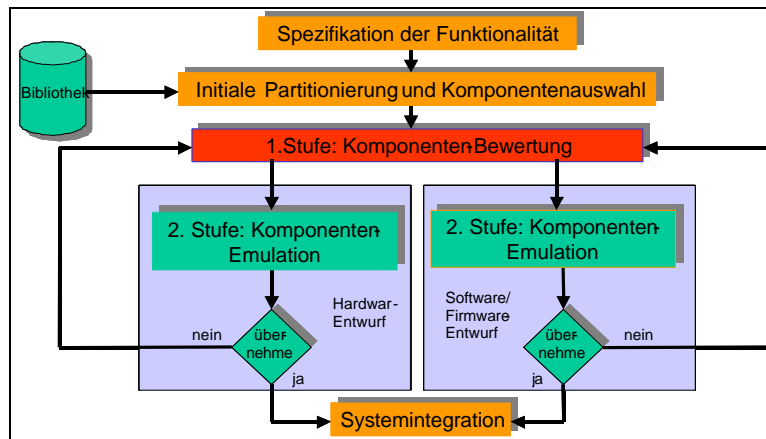


Abbildung 2:
Paralleler Entwicklungsablauf

Die Vorteile liegen darin, dass man sehr frühzeitig einen tiefen Einblick in das Systemverhalten der einzelnen Komponenten gewinnt. Der Aufwand zur Einführung von Alternativen ist wesentlich geringer, somit das gesamte Entwicklungsrisiko besser beherrschbar. Als Nachteil empfinden einige Entwickler den Aufwand zur Einführung einer entsprechenden Rapid Prototyping Umgebung. Eine tiefere Darstellung der Methodik ist in [We00] zu finden.

4.1.2.1 Toolset SPYDER

Betrachtet man die Struktur komplexer eingebetteter Systeme, so bestehen sie in der Regel aus einer anwendungsspezifischen Software und einer anwendungsspezifischen Hardware, wie in Abbildung 3 dargestellt. Die Software wird auf einem Mikrocontroller ausgeführt, welcher über eine Schnittstelle mit der Hardware kommuniziert. Diese kann entweder aus einem oder mehreren ASIC's bestehen oder in einem FPGA implementiert sein bzw. auch aus ASIC und FPGA bestehen. Das gesamte System steht zusätzlich in einem speziellen Kontakt mit der Umwelt.

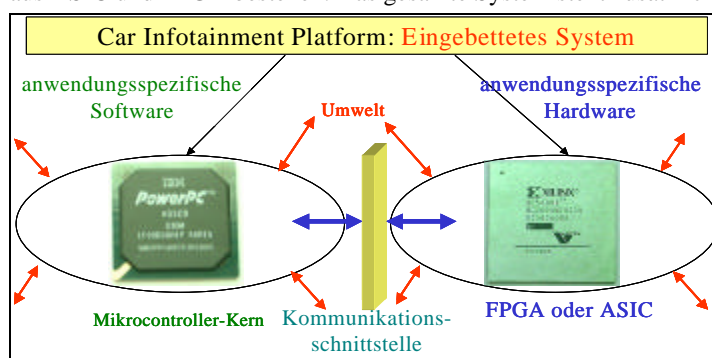


Abbildung 3:
Grundstruktur eines eingebetteten Systems

Daraus leiten sich die Anforderungen an eine entsprechende Rapid Prototyping Umgebung ab. Um Software-Komponenten (wie z.B. MP3-Algorithmen, MMI-Konzepte, Betriebssysteme) zu emulieren, wird ein Ziel-Prozessor-System zusammen mit seinem Speicher und Schnittstellen zur Entwicklungsumgebung (serieller Port, Ethernet) benötigt. Das verwendetet Tool-Set SPYDER stellt dazu geeignete Emulationsplattformen basierend auf IBM-PowerPC405 oder Hitachi SH3 zur Verfügung. Beide Mikrocontroller-Familien haben bereits im Bereich Car Infotainment eine gewisse Marktführerschaft übernommen.

Für der Entwicklung von anwendungsspezifischer Hardware stellt das Tool-Set SPYDER entsprechende FPGA-Boards zur Verfügung, welche in den PCI-Slot eines Entwicklungsrechners eingebaut werden. Damit kann ein Entwicklungsingenieur seine Hardware direkt an seinem Arbeitsplatzrechner emulieren und testen. Beide Teilsysteme können auch Stand-Alone zusammengesetzt werden. Damit kann die gesamte Zielsystemarchitektur getestet werden. Weitergehende Informationen über das Tool-Set SPYDER können unter [X2E00] gefunden werden. In Kapitel 5 wird die Anwendung der parallelen Entwurfsmethodik und der Einsatz von SPYDER demonstriert, eine ausführliche Darstellung in [We00] zu finden.

5 Rekonfigurierbare Car Infotainment Plattform

In dem folgenden Kapitel wird die Architektur der Car-Infotainment Plattform dargestellt, welche als eine integrierte Obermenge der zur Zeit verfügbaren Funktionen betrachtet werden kann. Sie dient als eine Art Referenzplattform, auf der verschiedenste Anforderungen für Fahrzeuge der Oberklasse bis in den Bereich der Kleinwagen evaluiert werden können, um eine sogenannte „Running Specification“ für den Fahrzeughersteller zu erarbeiten. Dabei soll auf der einen Seite die Bedeutung der Rekonfigurierbarkeit und auf der anderen Seite die Komplexität des Gesamtsystems erläutert werden. Diese Komplexität wird mit Hilfe der in Kapitel 4 dargestellten parallelen Entwurfsmethodik in entsprechend kleiner Teilprobleme zerlegt und parallel entwickelt.

5.1 Gesamtarchitektur

Die Gesamtarchitektur besteht aus zwei Leiterplatten. Das BASEBOARD ist Trägermodul für verschiedene Add-On Module und wird in Abbildung 4 dargestellt.

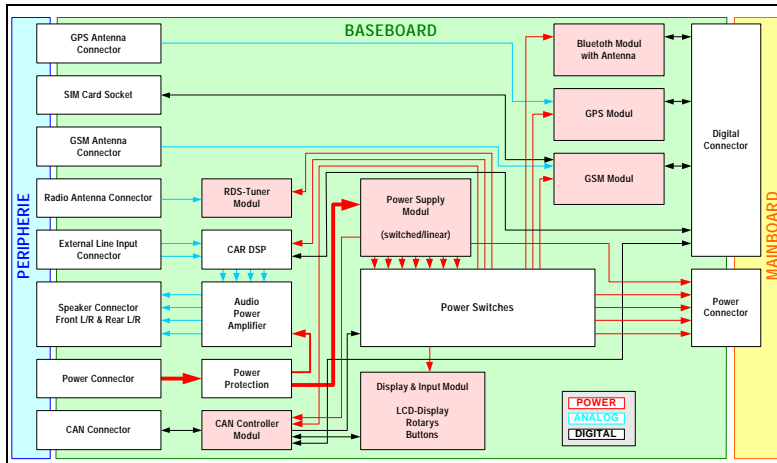


Abbildung 4:
Blockdiagramm BASE BOARD

Ein Power Supply Modul erzeugt aus der Kfz-Spannung bis zu 10 stabilisierte Einzelspannungen, welche über entsprechende Power Switches in zeitlich deterministischer Reihenfolge geschaltet werden. Ein komplettes Audio-System wird durch einen RDS-Tuner, CAR-DSP und Verstärker gebildet. Für einfache Anwendungen ohne aufwendige Graphik steht ein einfaches LCD-Display zur Verfügung.

Das GPS-Modul beruht auf einem Hochleistungs-DSP, auf dem in einer späteren Firmware-Version auch zusätzlich noch ein Sprachein- und Ausgabe-System sowie MP3-Kompression und Dekompression realisiert wird. Das GSM-Modul dient zum einen zum mobilen telefonieren, kann aber andererseits über eine entsprechende Software auf dem Hauptrechner-System des MAINBOARDS zum Internet-Zugang genutzt werden. Ein Bluetooth-Modul steht als drahtlose Schnittstelle für spätere Applikationen zur Verfügung.

Entsprechende Steckersysteme verbinden diese Grundinfrastruktur mit dem MAINBOARD, welches in Abbildung 5 dargestellt wird: Es wird für rechenaufwendige Anwendungen (MOST-Master, Navigation, 2-D Bildschirm-Farbgraphik, Mensch-Maschine-Interface) benötigt.

Der Kern des MAINBOARDS bildet ein Hauptrechnersystem basierend auf einem Embedded PowerPC405CR von IBM (266MHz/133MHz) mit 128MByte SDRAM Hauptspeicher, 128MByte Flash und einem 2-D Graphikcontroller. Ein 10/100Mbit Ethernet-Interface wird zur Kopplung an ein LAN benutzt, welches vorwiegend für die Software-Entwicklung eingesetzt wird.

Als zentrale Datendrehscheibe dient ein FPGA, in dem im wesentlichen Interface-Komponenten implementiert wurden, wie in Abbildung 6 dargestellt wird. Für die Ankopplung an den MOST-Bus (Media Oriented System Transport) wurde ein spezieller Hardware Adaption Layer (MOST-HAL) entwickelt, der den sogenannten parallel combined Mode unterstützt. Dadurch sind sowohl synchrone als auch asynchrone Datentransfers über einen gemeinsamen Transceiver-Chip (OS8104) möglich.

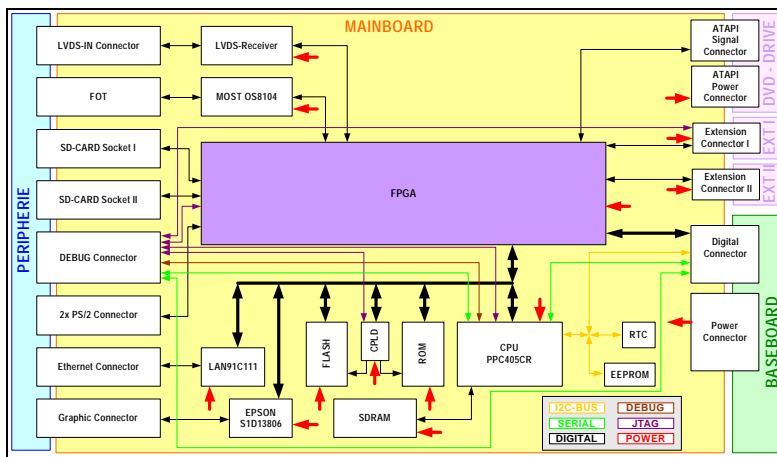


Abbildung 5:
Blockdiagramm MAINBOARD

Zur Anschluss eines DVD-ROM-Laufwerks sowie einer für Automotive zugelassenen Festplatte dient das ATAPI-Interface. Mehrere UARTs dienen als Verbindungsschnittstelle zu externen Modulen wie GSM/GPRS, GPS usw. Ein spezieller Hardware-Core unterstützt den Zugriff auf die beiden externen Secure Digital Cards (SD-Cards). Eine Wishbone-Bridge verbindet den FPGA-internen Wishbone-Bus mit dem externen PowerPC Mikrocontroller-Bus. Für bestimmte Internet-Funktionalitäten wird eine Tastatur mit Mouse gebraucht, welche über das doppelt vorhandene PS2-Interface (Tastatur + Mouse) angeschlossen werden kann. Zur Verwaltung der verschiedenen Audio-Quellen (I2S-Datenströme) wurde eine in Hardware realisierte Audio-Management-Unit integriert.

Insgesamt werden für diesen FPGA etwa 600k-Gatter Äquivalente benötigt. Der Konfigurationsdatenstrom wird nach dem Einschalten der Betriebsspannung vom PPC405CR aus einem Flash-Speicher in den FPGA geladen. Betrachtet

Tranceiver geschickt werden können. Dieser MOST-HAL wird noch zusätzlich über entsprechende große FIFOs vom FPGA-internen Bus entkoppelt, um die Interrupt-Belastung des Prozessors zu reduzieren.

Um diese Teilentwicklung durchführen zu können, wird nur das Hauptrechnersystem und die MOST-spezifische Hardware (HAL im FPGA, OS8104 und FOT) benötigt, welche auf einer Trägerplatine aufgebaut wurde. Dieses Zusatzplatine wird mit der Rapid Prototyping Plattform SPYDER-CORE verbunden, welche das Hauptrechnersystem zur Verfügung stellt. Diese Plattform hat den Vorteil, das alle Signale sehr effizient über ein hochintegriertes Steckersystem mit einem Speicher-Oszilloskop bzw. Logik-Analysator verbunden werden kann. Damit ist eine sehr schnelle und effiziente Firmware-Entwicklung möglich, wie in Abbildung 8 dargestellt.

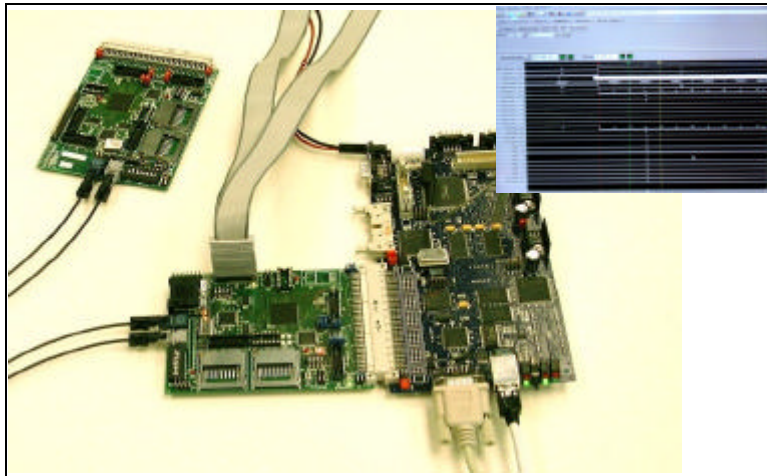


Abbildung 8:
Entwicklungsaufbau
MOST auf SPYDER-CORE

5.2.2 3-D Graphik Modul

Bei diesem Beispiel wird ein hochleistungsfähiger 3-D Graphikcontroller (Fujitsu-Orchid) für die Integration ins Gesamtsystem evaluiert. Er verfügt über externen Video-Speicher und hat neben dem Mikroprozessor-Interface noch einen weiteren digitalen Eingang (YUV), an dem ein Video-Decoder angeschlossen wurde, um auch analoge Videosignale einzuspeisen (z.B. von einer Rückfahrkamera). Mittels eines digitalen Eingangsmultiplexers, welcher in einem FPGA mit integriert wurde, kann sowohl ein Mikrocontroller (hier der PPC405) auf den Graphikcontroller zugreifen als auch ein digitaler Datenstrom (LVDS) angezeigt werden. Den Entwicklungsaufbau zeigt Abbildung 9.

Diese Architektur kann ebenfalls auf die Rapid Prototyping Plattform SPYDER-CORE abgebildet werden, wie in Abbildung 10 gezeigt wird. Dabei wurde der FPGA, der Graphikcontroller sowie die notwendigen Peripherie-Chips auf eine Trägerplatine gesetzt und an das Hauptrechnersystem auf der Plattform SPYDER-CORE angesteckt. Auch hier ist eine effiziente Firmware-Entwicklung möglich, da alle Signalleitungen über hochintegrierte Steckersysteme von einem Logik-Analysator messbar sind.

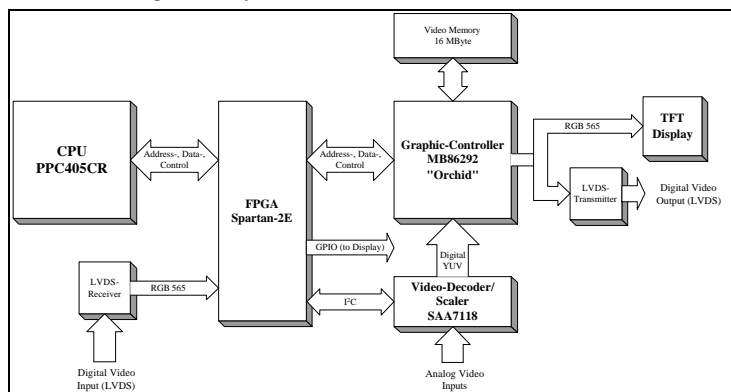


Abbildung 9:
Blockdiagramm 3-D-Graphikmodul

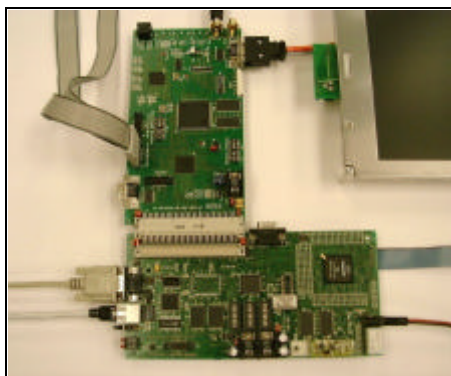


Abbildung 10:
Entwicklungsaufbau
3-D Graphik auf SPYDER-CORE

6 Fertiges Zielsystem

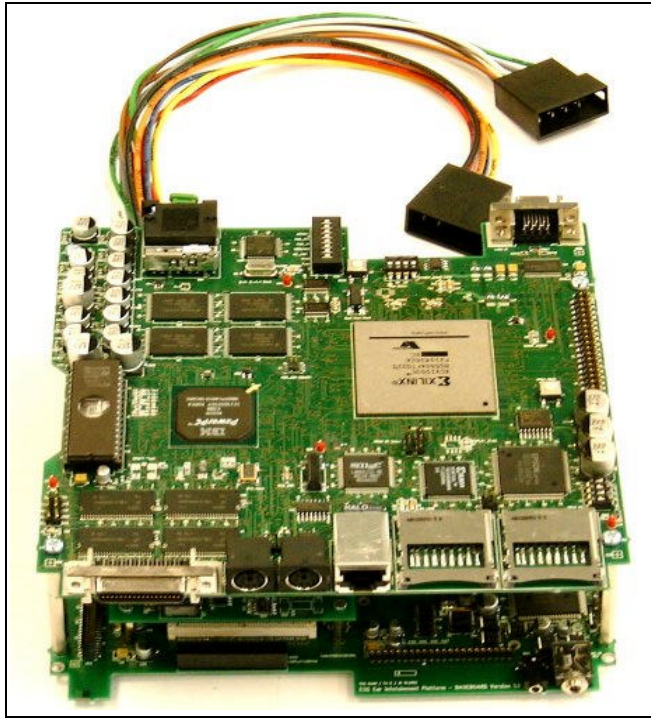


Abbildung 11:
CAR Infotainment Plattform Version 1.

Abbildung 11 zeigt das Zielsystem. Der entscheidende Unterschied zur in Kapitel 3 eingeführten sequentiellen Methode besteht darin, dass das eigentliche Zielsystem zum Abschluss des Entwicklungsprozesses entsteht, wobei das Zielsystem aus vielen einzelnen, parallel entwickelten Teilsystemen zusammengesetzt wird.

7 Zusammenfassung

Dieser Bericht zeigt die Vor- und Nachteile von FPGA- im Gegensatz zu ASIC-Implementierungen. Für das Anwendungsfeld Car-Infotainment erscheint zur Zeit eine FPGA-basierte Lösung mit weniger Risiko behaftet, da die Anforderungen an solche Systeme noch Gegenstand der Diskussion sind, sich somit stabile Spezifikationen noch nicht etabliert haben.

Es wurden verschiedene Implementierungsalternativen erläutert. Zur Zeit wird hauptsächlich die CTR-Alternative verwendet. Modernere Ansätze, welche zur Laufzeit die Konfiguration austauschen, also die gRTR oder auch pRTR führen zu einer zusätzlichen Steigerung bei der Ausnutzung von vorhandenen FPGA-Ressourcen und damit zu einer Kostensenkung bei der Implementierung von Funktionen auf einen gegebenen FPGA-Chip, wie im nächsten Kapitel gezeigt wird.

Basierend auf diesen Einführungen wurden Entwicklungsmethoden für komplexe eingebettete Systeme vorgestellt, die im wesentlichen in eine sequentielle und parallele Methode unterteilt werden können. In diesem Bericht wurde die parallele Methode benutzt, die wesentlich auf Rapid Prototyping basiert. Dazu wurde auch eine entsprechende Plattform mit Namen SPYDER vorgestellt. Entscheidendes Kriterium ist dabei, dass das eigentliche Zielsystem erst am Ende des Entwicklungsprozesses entsteht. Abschließend wurde die Methodik am Beispiel der Car Infotainment Plattform anhand von zwei Teilentwicklungen demonstriert.

8 Ausblick

Im Rahmen der Weiterentwicklung wird das MAINBOARD auf Basis der VIRTEX-II-Pro-FPGA-Technologie weiterentwickelt. Dabei wird der PPC405 mit ins FPGA-Chip integriert. Nach Leistungsbedarf kann auch ein zweiter PPC405-Rechnerkern im sogenannten Plattform-FPGA aktiviert werden. Diese Rechenleistung wird in Zukunft notwendig werden, wenn hochentwickelte Software-Konzepte wie JAVA und darauf aufbauend OSGi-Layers (Open System Gateway Initiative) angewendet werden.

Um die Ausnutzung vorhandener FPGA-Ressourcen weiter zu erhöhen, wird die pRTR angewendet. Dabei wird die partielle Rekonfigurierbarkeit des FPGAs ausgenutzt. Beispielsweise müssen Interface-Schaltungen nur dann verfügbar sein, wenn Zugriffe über sie stattfinden sollen. Daraus folgt, dass z.B. ein ATAPI-Interface gegen ein SD-Card-Interface partiell ausgetauscht werden kann, wenn auf die SD-Cards zugegriffen wird und umgekehrt. Wird nachfolgend das ATAPI wieder gebraucht, kann dieses wieder partiell installiert werden. Dadurch belegen beide die gleiche FPGA-Chip-Fläche und erhöhen so die Funktionalität pro Fläche ohne die Chip-Ressourcen (und damit auch die Kosten) zu erhöhen.

Literaturverzeichnis

- [Act02] ____: *General Purpose FPGAs*. <http://www.actel.com>
- [Xil98] ____: *The Programmable Logic Data Book*. Xilinx Inc, <http://www.xilinx.com>, 1998
- [Alt02] ____: *Flex 10k Embedded Programmable Logic Family Data Sheet*. <http://www.altera.com>, 2002
- [Xil02] ____: *Virtex-II Pro Platform FPGA Handbook*. Xilinx Inc, 2002
- [HuWi95] B.Hutchings, M.Wirthlin: *Implementation Approaches for Reconfigurable Logic Applications*. Proceedings of 5th International Workshop of Field Programmable Logic and Applications, Oxford 1995
- [WeKiRo98] K.Weiß, R.Kistner, W.Rosenstiel: *Analysis of the XC6000 Architecture for Embedded System Design*. IEEE Symposium on Field Programmable Custom Computing Machines (FCCM), Napa Valley CA, April 1998
- [We00] K. Weiß: *Architektorentwurf und Emulation eingebetteter Systeme*. ISBN 3-89722-393-7, Logos Verlag Berlin, 2000
- [X2E00] ____: *SPYDER-System User Manuals*. <http://www.x2e.de>, 2000