DFG-Schwerpunktprogramm
“Kooperierende Teams mobiler Roboter in dynamischen Umgebungen“

Aspekt:
“ Sensor-/Aktorinterfacing“

Thomas Ihme

1. Motivation und Ausgangssituation

Ein effizientes Sensor-/Aktor-Interface ist die Voraussetzung, um echtzeitgerechte Sensordatenströme liefern zu können. Dabei spielen Geschwindigkeiten und Zeitkonstanten aus der Umgebung des Roboters die entscheidende Rolle. Je schneller Aktionen und Reaktionen im Spiel (oder jeder anderen beliebigen Applikation) sind, um so größer ist die Dynamik. Um die Dynamik der Prozesse zu erfassen, muss die Abtastrate mindestens doppelt so groß sein wie die maximal auftretende Frequenz. Anders ausgedrückt: Es muss mindestens zweimal in einem Zeitraum der kürzesten vorkommenden Zeitkonstante abgetastet werden (Shannon-Theorem). Das bezieht sich natürlich auf den zu beobachtenden Prozess.

Da es nur möglich ist, auf Ereignisse reagieren zu können, die auch beobachtet werden können, müssen Beobachter (Sensoren) für die betreffenden Ereignisse und Prozesse vorhanden sein und eine ausreichende Datenrate von Sensorinformationen bereitstellen. Entsprechendes gilt für Aktoren.

Bei mobilen Robotern besteht das besondere Problem darin, dass eine Vielzahl von Sensoren und Aktoren auf engem Raum vorhanden sind. Dazu zählen äußere Sensoren (Umgebungserkennung) und innere Sensoren (Antriebsregelung, Manipulator, Kamerasteuerung). Eine Übliche Methode ist, Mikro-Controller mit passender Peripherie zu nutzen. Bei einer Vielzahl von Sensoren/Aktoren ist dann ein ganzes Netzwerk von Controllern notwendig. Es ergeben sich dadurch folgende Probleme:

- Mikro-Controller müssen mit ihren Peripherie-Komponenten speziell programmiert werden, um die gewünschte Funktion zu erfüllen. Der Code ist dann für den Mikro-Controller spezifisch (schlechte Portabilität). Erschwerend sind schlechte Debug-Möglichkeiten.
- Die Peripheriekomponenten von Mikro-Controllern sind fest verdrahtet. Einschränkungen und Fehler im Design sind schwer mit Software zu kompensieren (z.B. fehlende Doppelpufferung bei Datenübertragungen)
- bei vielen Sensoren ist ein Netzwerk von Mikro-Controllern notwendig. Sensorinformationen müssen dann in der Regel durch das Netzwerk transferiert werden. Dabei entsteht durch die Kommunikation zusätzlicher Protokollaufwand.
- Bei hohen Sensorinformationsströmen oder komplizierten Signalverarbeitungsalgorithmen ist es möglich, dass ein einzelner Mikro-Controller mit einer einzelnen Aufgabe ausgelastet ist.

2. Sensor-/Aktor-Interface mit FPGA

Speziell für den Fall eines mit vielen Aktoren versehenen Laufroboters entstand das Konzept, die Peripheriesteuerung zu konzentrieren. Es war Ziel, die Peripheriesteuerung als leistungsfähiges Sensor-/Aktor-Interface mit einem leistungsfähigen Prozessor zu koppeln. Damit sollten sensorgeführte Bewegungen ermöglicht werden, die auf kurze Control-Zyklen angewiesen sind (ca. 1ms).
Beispielsweise sollen bei einer Gelenkregelung (Motor) die Position mit einem Potentiometer, die Drehzahl mit einem Inkrementalgeber (shaft encoder) und der Motorstrom erfasst werden, sowie die Motoransteuerung Puls-Weiten-moduliert erfolgen. Damit ergeben sich 3 Input-kanäle (2x analog, 1xZähler) und 1 output-Signal (Zähler/Vergleicher). Für eine Vielzahl von Gelenken (6 Beine, 3 Gelenke je Bein) ergeben sich entsprechend viele Sensor-/Aktor-Kanäle zuzüglich weiterer externer Sensoren (Infrarot, Kraftsensoren).

Die für ein solches Interface notwendige Funktionalität lässt sich in einen FPGA integrieren. Diese Lösung weist gegenüber der Mikrocontroller-Variante folgende Vorteile auf:

- *Entkopplung von CPU und Peripherie*
  Durch die Entkopplung von CPU und Peripherie wird die CPU von Peripheriefunktionen entlastet (z.B. Signalformung). Ein FPGA kann so konfiguriert werden, dass er selbständig zeit- oder ereignisgesteuert Sensorsignale ausliest (ADU, Zähler) und zwischenspeichert bzw. Aktoren-Steuersignale formt (PWM). Durch die Entkopplung von CPU und Peripherie werden die Anforderungen an Reaktionszeiten entschärft.
  Durch Wegfall der Bindung an spezielle Peripheriekomponenten von Mikro-Controllern besteht eine bessere Auswahl an leistungsfähigen Prozessoren. Auch ein Generationswechsel wird dadurch begünstigt.

- *Konzentration*
  Die Ein- und Ausgabefunktionalität wird auf engem Raum konzentriert. Dadurch entfällt die Notwendigkeit, Ein-/Ausgabefunktionen über mehrere Mikro-Controller zu realisieren und somit die Notwendigkeit der Kommunikation zur Sensordatenakquisition.

- *Modularität, Skalierbarkeit und Flexibilität*
  FPGA's (z.B. Xilinx Spartan-Serie) lassen sich mit Hilfe von hierarchisch aufbaubaren Logikplänen, Zustandsautomaten und VHDL programmieren. Einzelne Module (z.B. PWM-Generatoren) lassen sich wie Software wiederverwenden und entsprechend den Anforderungen zu komplexen Peripheriecontrollern skalieren. Durch die Programmierung der Logikfunktionen ist es möglich, auch nach Aufbau der Hardware Veränderungen vorzunehmen.

- *leistungsfähiges Interface zur CPU*
  In den FPGA lässt sich ein Interface integrieren, auf das wie auf RAM zugegriffen werden kann, so dass sich Memory-mapped I/O realisieren lässt. Damit entfallen Kommunikationsprotokolle und durch parallelen Zugriff wird die Übertragungsgeschwindigkeit erhöht.

Beispiel der Einbindung eines FPGA für ein Sensor/Aktor-Interface

Die Einbindung des FPGA könnte so erfolgen, dass die Baugruppe als PC104-Modul ausgeführt wird und somit eine standardisierte Schnittstelle vorhanden ist. Die Rechnerseite kann dann beispielsweise ähnlich dem Kurt2-Hauptrechner aufgebaut werden. Zur Anbindung der Aktorik sind Schalt-Leistungsverstärker notwendig. Die PWM-Signale werden durch den

FPGA erzeugt. Zur Verwendung von Analog-Sensoren werden ADU-Kanäle gemultiplext (Steuerung durch FPGA), so dass eine Vielzahl von Analogsensoren angeschlossen werden können. Für Inkrementalgeber zur Bestimmung der Drehzahl bei Motoren werden Quadratur-Dekoder und Zähler implementiert. Zusätzlich können Binär-Sensoren angeschlossen werden. Es ist möglich, den FPGA mit Hilfe eines seriellen Interfaces zu programmieren, so dass die Konfiguration der Logik des FPGA durch die CPU vorgenommen werden kann
Das folgende Bild zeigt eine mögliches System:

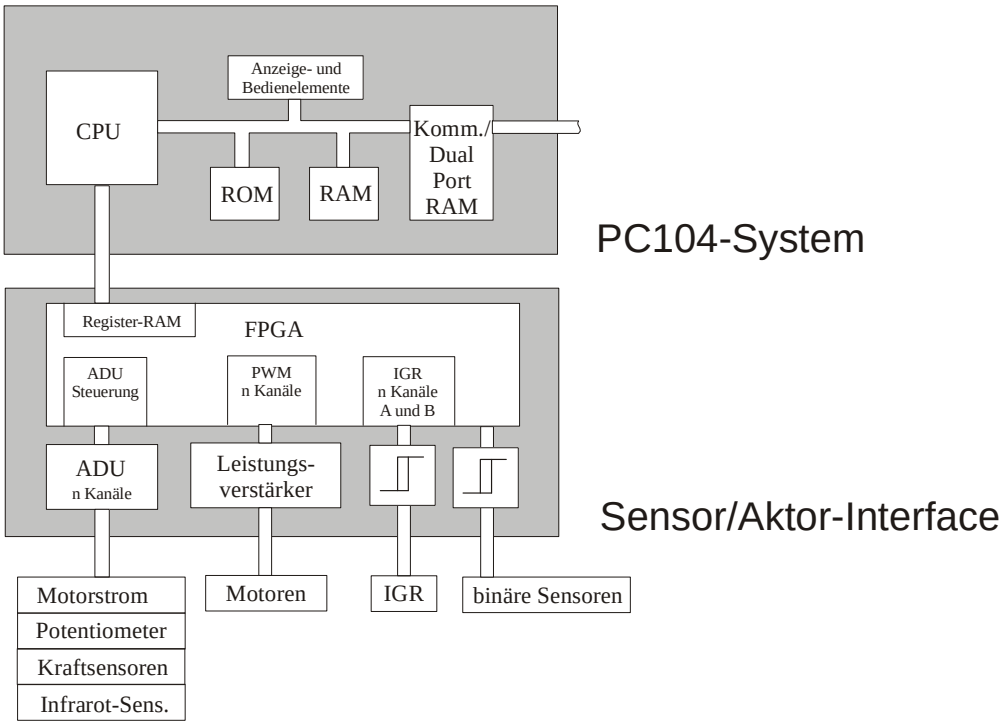

Bild 1: Beispielkonfiguration eines Sensor/Aktor-Interfaces mit FPGA

Konfigurationsbeispiel:
64 Analogkanäle
20 PWM-Kanäle
20 IGR-Kanäle