Projekt Neuronale Hardware

Kurzbeschreibung

Stand: 5. 5. 1998
1. Änderung: 3. 6. 1998

Ziel
Aufbau und Erprobung einer Hardwarestruktur zur Implementierung neuronaler Paradigmata der Informationsverarbeitung.

Ansatz
Nutzung zuhandener Mikrocontroller (μCs). Verwirklichung des naheliegenden Gedankens "1 Neuron = 1 μC".

Verbindungssystem
Ringbus auf Schieberegister-Grundlage. 1 Ring-Shift ermöglicht Verbindung "jeder mit jedem". Jedes Neuron "sieht" die Ausgangsbelegungen aller anderen Neuronen und legt nach jedem Schiebezyklus seine neu errechnete Ausgangsbelegung auf den Bus. Die logische Organisation beruht auf Eimern fester Länge (vergleichbar etwa mit den Zellen bei ATM) und gestattet es, neben zeitstarren auch beliebig variable Verbindungsschemata zu implementieren.

Systemstruktur

Fig. 1 zeigt die Systemstruktur.

Fig. 1 Systemübersicht

Erklärung:

1) Knoten. Jeder Knoten ist eine komplette Mikrorechneranordnung aus Controller (μC), Speicher und Steuerhardware. Weitere Einzelheiten in Fig. 2.
2) Interface (Schieberegister-Ringbus),
3) zentrale Bussteuerung,
4) Bedien- und Steuerrechner (PC).

Fig. 2 Struktur eines Knotens

Erklärung:

1) Mikrocontroller (Prozessor),
2) Arbeitsspeicher (SRAM),
3) Debugging-Hilfsspeicher (SRAM),
4) Interfaceanschluß- und Steuerschaltungen (ein einziger CPLD- oder FPGA-Schaltkreis),
5) Interface (Schieberegister-Ringbus; enthält auch Signale der "Infrastruktur", wie Takt, Rücksetzen, Fehlersignalisierung usw.),
6) Flash-ROM für Selbsttest- und Startprogramme (falls entsprechende IPL-Lösung gewählt).

Mikrocontroller
System Hitachi H8/300H steht fest. Es ist der kostengünstigste Schaltkreistyp einzusetzen.

Auswahl:

- der Typ mit dem kleinsten Flash-ROM,
- H8/3002,
- H8/3001 (erfordert Adreßdecodierung im Steuerschaltkreis).

Taktfrequenz:
Möglichst 20 MHz. Wenn kostengünstiger, dann auf 16 MHz zurückgehen.

Speicher

Arbeitsspeicher: wenigstens 128 kBytes. Im gegebenen Kostenrahmen so groß wie möglich auslegen. Anordnung z. B. $64k \cdot 16 + 64k \cdot 1^{*)}$ oder $2 \cdot 128k \cdot 8 + 256k \cdot 1^{*)}$.

*): Debugging-Hilfsspeicher.

Interface

Infolge der Ringstruktur handelt es sich um Punkt-zu-Punkt-Verbindungen von wenigen cm Länge, so daß besondere Treiberstufen unnötig sind.

Sonderprobleme (deren Lösung sich auf die Leiterplattengestaltung auswirkt):

1. Anpassungsmaßnahmen (z. B. Serienwiderstände): betrifft jene Leitungen, die Leiterplatten-Grenzen überschreiten,
2. gemeinsame Rückmeldungen (die von mehreren Knoten gleichzeitig erregt werden können). Lösungen: (a) Einzelsignale zur zentralen Bussteuerung, (b) Einbeziehung in die Paketstruktur, (c) Daisy-Chain-Verschaltung, (d) Wired OR. Wird letztere Lösung gewählt, sind ggf. zusätzliche Treiberstufen zu setzen.

Organisation des Schiebeweges:

- Länge des einzelnen Paketes: 72 Bits,
- Schieben der Pakete: durch Hardware,
- Abholen und Füllen der Pakete: durch Software,
- Schieben (wahrscheinlich) mit 6 Bits + 1 Parity-Bit parallel. Längs-Parität wird noch untersucht.
- Schiebefrequenz: 10...20 MHz,
- das Interface umfaßt insgesamt ca. 16 Signale.

Interfaceanschluß und Steuerschaltungen

1 programmierbarer CPLD-Schaltkreis (Flash). Muß in der Schaltung programmier- und löschbar sein (Boundary-Scan-Weg).

Die Anzahl der (Makro-) Zellen ergibt sich aus der Paketlänge (72 Bits) + ca. 20 für den Adreßvergleich + ca. 10 für allgemeine Steuerzwecke (Debugging-Betriebsarten, State Machines usw.). Insgesamt also etwas über 100 Flipflops.

Anschlußzahl: 32 (Interface) + 36 (Adresse + Daten) + 20 (div. Steuersignale) = ca. 88.

Realisierung z. B. Xilinx 9500 oder Lattice ispLSI.

IPL-Prinzip

IPL = Initial Program Loading = Laden der Software nach dem Einschalten. Die eigentliche Anwendungs- und Systemsoftware muß ladbar sein (keine Ausführung aus ROM). Zum Anfangsladen stehen folgende Lösungen zur Wahl:

1. Mikrocontroller mit eingebautem (Flash-) ROM,
2. zusätzlicher ROM (Pos. 6 in Fig. 2),

3. Speicherzugriffsweg über das Schiebe-Interface (wobei der Controller im Hardware-Standby-Modus gehalten wird). Erfordert bidirektionale Adreßbusanschlüsse im Steuerschaltkreis. Vorteile: (1) es ist alles "soft" (= leicht änderbar), (2) wir können ROM-lose Controller einsetzen.

Mechanischer Aufbau

Leiterplatte im Einfach-Europa-Format, 2-lagig. Steckverbinder DIN 4162, 64- oder 96-polig. Möglichst 4 Knoten gem. Fig. 2 unterbringen.

Auch Doppel-Europa-Format wäre zu untersuchen (mit 6...10 Knoten).

Die Knoten einer Leiterplatte bilden einen Teil der gesamten Ring-Anordnung (es führt nur ein Schiebeweg in die Leiterplatte hinein und einer heraus).

Bypass-Vorkehrungen (zum Umgehen einer defekten bzw. fehlenden Leiterplatte im System): werden noch untersucht. Erfordern u. U. zusätzlich 2 QuickSwitch-Multiplexer auf der Leiterplatte.

Entkopplung: wird noch untersucht. Ggf. Bustreiber für Leitungen, die nicht in den Ring einbezogen sind (Alternative: Einbeziehung aller Leitungen in den Ring (Daisy Chain)).

Hot Plugging/Live Insertion: nicht erforderlich.

Optimierung: wichtig sind geringste Gesamtkosten (Kosten/Leiterplatte · Anzahl der Leiterplatten → Minimum); die physische Größe spielt praktisch keine Rolle.

EMV-Vorkehrungen
Die gängigen EMV-Regeln beim Leiterplattenentwurf sind einzuhalten. Ausgiebige Simulation und Test sind nicht gefordert.

Neuronale Hardware (Projekt NHW)

Vorläufige Disposition

Stand: 3. 12. 1997

Ziel

Aufbau und Erprobung einer Hardwarestruktur zur Implementierung neuronaler Paradigmata der Informationsverarbeitung.

Ansatz

Nutzung zuhandener Mikrocontroller (µCs). Verwirklichung des naheliegenden Gedankens "1 Neuron = 1 µC".

Struktur des Neuronennetzes

Keine Beschränkung seitens der Hardware. "Jeder-mit-jedem"-Verbindung vorgesehen.

Verbindungssystem

Ringbus auf Schieberegister-Grundlage. 1 Ring-Shift ermöglicht Verbindung "jeder mit jedem". Jedes Neuron "sieht" die Ausgangsbelegungen aller anderen Neuronen und legt nach jedem Schiebezyklus seine neu errechnete Ausgangsbelegung auf den Bus. Die logische Organisation beruht auf Eimern fester Länge (vergleichbar etwa mit den Zellen bei ATM) und gestattet es, neben zeitstarren auch beliebig variable Verbindungsschemata zu implementieren.

Betriebssteuerung

Zentralisiert durch angeschlossenen PC. Zunächst Kopplung über Parallelschnittstelle, später (Endzustand) eine zugeordnete PC-104-Konfiguration (die ihrerseits z. B. über Ethernet anderweitig verkoppelt werden kann).

RAS-Vorkehrungen (Reliability, Availability, Serviceability)

Die diesbezüglichen Probleme einer Anordnung aus einer sehr großen Anzahl von µCs werden konstruktiv gelöst. Software in RAM (mit Ausnahme des Anfangsladens). Debugging-Vorkehrungen durch zusätzlichen Vergleichsstop-RAM (ermöglicht auch Single Step), entsprechende Busprotokolle und Systemsoftware. Selbsttestvorkehrungen. Eventuell Paritätsprüfung auf dem Schiebeweg. Keine ausgedehnten Tri-State- oder Wired-OR-Strukturen.

µC-Auswahl

Der kleinstmögliche (kostengünstigste) Schaltkreis mit ausreichender Leistung, genügend Adreßraum (> 64 kBytes), genügend I/O-Ports und möglichst flexibler Befehlsliste (+ Adressierungsmodi). µC muß (1) preisgünstig sein und (2) sowenig Außenbeschaltung (Glue Logic) wie möglich erfordern. Die 8051-Leistungsklasse leistet zuwenig. PIC 17Cxx ist

programmseitig nicht flexibel genug. SAB 16x hat weniger geeignetes Adressierungsschema. Intel ..86 kommt gar nicht in Frage. Motorola 68 000 dürfte wegen der Anforderung an die Außenbeschaltung ausscheiden (wäre aber zu prüfen - bes. 68EC000 oder 68330). (Es sollte sich um einen echten Controller handeln - mit internem Speicher und genügend flexibel nutzbaren I/O-Ports, nicht um einen reinen Prozessor.) Bevorzugt: Hitachi H8/300H.

Technischer Aufbau

Leiterplatten in Rahmen. Leiterplatten in Einfach-Europa-Format (100 * 160 mm) mit jeweils 4 µC-Knoten. *Prüfen:* Doppel-Europa-Format mit 8 oder mehr Knoten (Kostenoptimierung). 1 Knoten = µC + 128/256 kBytes RAM (einschl. Debugging- und ggf. Steuer-Zusatz-RAM) + Busanschaltung + (falls notwendig) "Restlogik". Evtl. die gesamte Zusatzlogik in einem CPLD/FPGA je Leiterplatte. Zentrale Taktzuführung. Entwurfsentscheidung synchrones/asynchrones Timing der Kommunikation ist noch zu treffen!

Anzahl der Neuronen bzw. µCs

Maximal wenigstens 1k = 1024. Die Beschränkung betrifft:

1. die praktische Ausgestaltung. 1k Knoten = 256 Leiterplatten = 1 Schrank = Kosten weit über 100 000 DM.
2. die Verarbeitungsleistung. Wie lange dauert ein System-Zyklus? (Bildung des Skalarproduktes über Vektoren mit 1k Elementen + Kommunikationsdauer).
3. die vorzusehenden Adressierungsstrukturen. Vorläufige Festlegung: Adreßfelder von 10 Bits für 1k Knoten. Aber so entwerfen, daß jederzeit "verlängert" werden kann.

Erste praktische Implementierung: mit ca. 40 Neuronen = 10 Leiterplatten = (vermutlich) um 10.000 DM Kosten.

Multitasking

Es liegt nahe, auf einem µC-Knoten mehrere Neuronen "rechnen" zu lassen. . Mit 8 Neuronen je Knoten bedeuten 1k Neuronen = 128 Knoten = 32 Leiterplatten = 1 19"-Rahmen = (vermutlich) unter 50 000 DM Kosten. Ausführung aber erst, wenn Grundkonfiguration (1 Neuron = 1 µC) funktioniert.

Wissenschaftlicher Nutzen

1. als Forschungsgegenstand an sich: Darstellung einer "extremen" Multiprozessorkonfiguration, Lösung der spezifischen Detailprobleme (Kommunikation, Softwareentwicklung, Debugging, RAS, Betriebszuverlässigkeit),
2. Plattform für Forschungsarbeiten zu neuronalen Netzen,

3. Plattform für Forschungsarbeiten zu “massiv parallelen” Algorithmen für an sich beliebige Anwendungen,
4. Plattform zu anderweitiger Nutzung (z. B. zur Schaltungssimulation).

Entwicklungsschritte (Übersicht)

- Einarbeitung in Problematik “neuronale Netze”
- µC-Auswahl
- Bestimmung des Speichersubsystems
- Bestimmung des Kommunikationsprotokolls
- Aufbau einer Einzelprozessor-Testplattform: Wrap-Technologie, direkte PC-Verbindung über Parallelschnittstelle, Dual-Port-RAM. Zweck: Vorerprobung von Hard- und Software.
- technische Entwicklung (Leiterplatte, Rahmen, Interface zum Steuer-PC)
- Architekturentwicklung
  - Kommunikationsprotokolle
  - allgemeine Wirkprinzipien (Theory of Operation)
  - RAS-Vorkehrungen
  - Industriestandard-Schnittstelle(n) (PCs, Netzwerke usw.)
- Software
  - Hardwaretest und IPL (Download)
  - Debugging
  - Betriebssystemkernel
  - Anwendung (Neuronennetz-Implementierung)
  - Testbeispiele und Verifizierung

PERT-Diagramm wird noch ausgearbeitet.

Zur Testplattform (Fig. 1)
Einfachster Aufbau zum Prüfen der Grundfunktionen und als Plattform zur Software-Inbetriebnahme. µC wird "ROMless" betrieben. Download über DP-RAM vom PC aus. Erhöhter Aufwand zur Adreßdecodierung usw. spielt keine Rolle.

Festzulegen: Speicherdatenweg 8 oder 16 Bits breit. (Bevorzugt: 8 Bits; später leistungsentscheidende Unterprogramme in den µC-internen ROM).

RAM erhält zusätzliche Bits (wenigstens eines für Vergleichsstop, evtl. weitere für Wartezustandssteuerung usw.

Zu prüfen: wie sollen Knoten endgültig aussehen? - Kostenfrage: (1) µC-Kosten, (2) Zusatzbeschaltung (RAM usw.), (3) RAS (Fehlersuche, Software-Änderungen).

Alternativen:
1. alle Knoten haben externe RAM mit Debugging-Vorkehrungen
2. Knoten ohne externen Speicher, internen Flash-ROM nutzen
3. ein Knoten auf jeder Leiterplatte hat "alles" (externen RAM + Debugging-Features), die anderen sind reine "Single Chip"-Controller; Software, in der Fehler gesucht werden soll, wird auf diesem Knoten ausgeführt
4. die gesamte Anordnung ist SIMD (alle µCs einer Karte arbeiten vollsychron auf einen Speicher) - was allerdings die Universalität beachtlich einschränken dürfte.
5. zunächst wie 1. bauen (Zuhandenheit). Endlösung enthält Controller eigener Architektur mit eingebauten Debugging-Vorkehrungen.

Zum Kommunikationssystem
Schiebeweg, vorzugsweise auf Grundlage Boundary Scan (JTAG/IEEE 1149). Boundary Scan ist aber kein Dogma (sondern wird nur wegen Zuhandenheit in Erwägung gezogen). Wenn es nicht paßt: CPLD, z. B. Xilinx 9500. Herkömmlicher Schiebeweg. Evtl. eine CPLD für 2 oder 4 Knoten. (Nimmt auch die weitere Restlogik auf.) Evtl.mit 2...4 Bits parallel schieben.

Zu prüfen:

- Folge der Schiebe- und Verarbeitungsschritte
- Synchronisation, Fertig- und Fehlermeldungen
- rein bitserielle oder in mehreren Bits paraller Schiebeweg?

Es genügt, wenn der Schiebeweg so schnell ist, daß die µCs mitkommen. Schieben mit 10 MHz ist unproblematisch.

Elementare Informationsstrukturen

Zellen mit fester Anzahl an Bits (Eimer). Prinzip wie z. B. ATM. Aber Eimer so klein wie möglich (Schiebezeiten, Aufwand).

Ein Eimer sollte eine Knoten-Adresse (= 10 Bits) + genügend Steuer-Information aufnehmen können.

1. Versuch: 18-Bit-Eimer gem. Fig. 2

18 Bits entsprechen einem 18-Bit-JTAG-Bustreiber (TI Widebus o. ä.).

Übertragungsrichtungen
Bit 16 gibt die Übertragungsrichtung an:

1. Empfangen (RCV): Information ist für µC bestimmt
2. Senden (XMIT): Information wurde von µC aufgeschaltet.

In Betriebszustand "Ausführen" (Executive State) hat Bit 16 eine abgewandelte Bedeutung: Kennzeichnung Alt-Neu (ein Knoten, der fertig ist und seinen Ausgangswert aufgeschaltet hat, invertiert Bit 16). Eine zusätzliche Ringleitung (zentral gesteuert) gibt an, welche Belegung im nächsten Umlauf als "neu" anzusehen ist (Odd/Even Executive Cycle).

2 Formate:

a) kontextfrei. Der Inhalt wirkt "für sich", ohne Bezug auf zuvor übertragene Information.
   - Empfangen = Kommandoübertragung. Eimer enthält Knotenadresse + 6 Bits Kommandocode (reicht, um 64 verschiedene Kommandos zu codieren).
   - Senden = Statusübertragung. Eimer enthält Knotenadresse + 6 Bits Zustandscode (reicht, um 64 verschiedene Zustandsmeldungen zu übertragen).

b) kontextabhängig. Der Inhalt umfaßt 1 16-Bit-Wort bzw. 2 Bytes. Was diese bedeuten, häng vom jeweiligen Zustand ("Kontext") des Knotens ab. Da hier keine Knotenadressen enthalten sind, kann das Format nur genutzt werden, wenn - unter zentraler Steuerung - gewährleistet ist, daß jeder Knoten "seinen" Eimer vorfindet.

Grundkommandos:

- nichts tun (NOP)
- Globalsteuerung (Broadcast): wirkt auf alle Knoten. Feld DVC ADRS enthält nähere Angaben (wirkt hier als Opcode).
- Steuern (Control): überträgt weitere Parameter der Kommandoausführung (z. B. Speicheradressen. Lämgenangaben usw.)
- Abfragen (Sense). Holt genauere Statusangaben aus dem Knoten ab.
- Schreiben (Write): überträgt Datenstrom zum Knoten.
- Lesen (Read): holt Datenstrom vom Knoten.
- Ausführen (Execute): "neuronale Verrechnung". Knoten empfängt Eingangswerte bzw. legt Ausgangswerte im Eimer ab.

Problem:

Die einzelnen Zustände des Kommunikationswegs in weiteren Bits oder über gesonderte Leitungen signalisieren?

Zusätzliche Signale (außer Takt usw.):

Rücksetzen (Reset)
Ausführungszustand (Executive State)
Zykluskennzeichnung im Ausführungszustand (Odd/Even Executive Cycle)

Wenn JTAG nicht genommen wird: evtl. auf 20 Bit-Eimer gehen. Kann Knoten-Adresse (DVC Adrs; 10 Bits) + 1 Byte Information + 4 Steuerbits aufnehmen. 4 Bits parallel schieben.

Noch zu lösen:

Besetzt- bzw. Fertigmeldung der einzelnen Knoten. Wired OR liegt nahe, hat aber RAS-Nachteile (Fehlersuche, Herausfinden "schuldiger" Knoten).

Ansätze:

- doch Wired OR, zusätzlich Polling über den Schiebeweg,
- schnellerer Extra-Schiebeweg (je Knoten ist nur 1 Bit erforderlich; 4 könnten ohne weiteres parallel verschoben werden. 1k Knoten = 256 Schiebetakte = 25 µs.
- kombinatorisches Daisy Chain, auch mit 4 Bits auf einmal. 1k Knoten = 256 zu durchlaufende Gattereben zu 50 ns (pessimist. Annahme) = 12,5 µs (oder schneller).

Grundsätzliche Alternative:

Größere Eimer, die grundsätzlich Quelle und Bestimmung enthalten (Anregungen gem. ATM oder Token Ring Netzwerk). Bringt womöglich mehr Flexibilität (= Narrenfreiheit). Dann müßte aber wohl auf jeden Fall mit mehreren Bits parallel geschoben werden.

Einflußgrößen: Länge der Schiebekette, Hardware-Aufwand, belegte Pins am μC.

2. Versuch (16. 11. 97):

Eimer müssen Source- und Destination-Angaben aufnehmen können. 2 * 10 = 20 Bits. Bedeutet generell Übergang auf 32...36 Bits (evtl. 34 gem. bish. Schema (Fig. 2)).

- in den Prozessor z. B. jew. 16/18 Bits übernehmen.
- im Executive State ggf. nur 16 Bits übernehmen, Rest (wenn nicht kontextfrei) vernachlässigen (einstellbarer Übertragungsmodus für Rechnen mit geringerer Genauigkeit)
- andererseits mehr als 16 Bits ohnehin günstiger, da System hierdurch vielseitiger wird.

Eimer-Inhalte:

a) 20 Bits Source/Destination + Grundkommando + 1 Kommando-, Status- oder Datenbyte (Multiplexübertragung),
b) 10 Bits Source + ≥ 16 Bits Daten (Burst-Mode; 1 Knoten kann zu einer Zeit genau einen solchen Datenstrom steuern),
c) alles Daten (globaler Burst Mode), den jeder Knoten anfordern kann (im Ggs. zum Executive State wird der Datenstrom von allen Knoten, die nicht Source oder Destination sind, ignoriert),
d) alles Daten (Executive State).

1. 12. 97:

Aufbau Knoten:

H8/3003-Prozessor (elementare ROM-Ausstattung). Typ wird noch genau bestimmt (Flash oder OTP); auf Steckkarte (in Prüfadapter) oder im System programmierbar. ROM nimmt nur Anfangslader usw. auf (d. h. "sichere" Programmstücke, an denen nicht dauernd zu ändern ist). Flash-Programmierung kostet Zeit, und es kann sein, daß es nicht möglich ist, alle Schaltkreise im System parallel zu programmieren.

Speicher: 128k*8 + 256k*1 oder 64k*16 + 64k*1

Restlogik: möglichst in 1 CPLD. Z. B. Xilinx XC9572 oder 95108.

Eimergröße: 36 Bits.

Schieben: 6 Bits parallel + Paritätsbit

1 Adreßbereich des Prozessors dient zum Zugriff auf Schiebedaten (16 Bits breit).

2 Arten der Synchronisation:

a) Interrupt: nach Schieben wird, falls gültige Daten anliegen, ein Interrupt ausgelöst
b) Wartezustand: eintreffende Schiebedaten führen CUP aus Waitstate heraus (im Executive State)

Der Empfang von Schiebedaten muß stets bestätigt werden. Zentrale sieht nach, ob auch alles bestätigt worden ist. Empfangene Daten werden entweder nur bestätigt oder (bei Vermerk der Bestätigung) durch neue ersetzt. Daten, die nicht bestätigt wurden, werden immer wieder angeboten. Ganz exakt: der ursprüngliche Sender muß warten, bis der jeweilige Eimer mit Bestätigung zurückkommt und diesen entweder ganz vom Bus nehmen (d. h. als frei kennzeichnen) oder neu belegen. Prinzip ähnlich Token Ring (was, um etwas zu lernen, ausgewertet werden sollte).

Entwurf Okt. 98
Versuch Schiebering neuron. Hardware:

Signale:
Shift CLK, Strobe, Reset

Zustände: executive, non-executive, IPL

im Executive State 32 + 4 Bits

IPL: 36 Bits 16 + 16

im Non Executive State 72 Bits

32 Daten + 32 Zieladresse (16 + 16)

32 = 2 * 16 Ziel/Quelle + 16 Adresse in Ziel + 16 Daten