Miniprojekt: Minimax-Maschine
Pflichtenheft

Maximilian Lumpe, Niklas Blume, Jan Feuchter, Phu Bac Duong

Hardware Projekt 2017
03. Januar 2017

Contents

1 Einleitung

In diesem Miniprojekt im Rahmen des Hardware-Praktikums beschäftigen wir uns mit der Minimax-Maschine, welche uns grundlegend aus der Vorlesung "Grundlagen der Rechnerarchitektur" bekannt ist. Zur Lösung der Aufgaben ist es hierbei notwendig, die vorgegebene Grundstruktur der Maschine geeignet zu erweitern, um die Algorithmen zu realisieren.
Die Vorbereitung auf unser Projekt wird dokumentiert und strukturiert durch das von uns erstellte Pflichtenheft. Das Pflichtenheft wird nur unsere Vorbereitung beinhalten. Die Ergebnisse werden in einer weiteren Dokumentation enthalten sein.

2 Aufgabe: Paketanalyse

Nach unserem Verständniss ist das Ziel der Aufgabenstellung das Implementieren des Algorithmus "Paketanalyse" auf der Minimax-Maschine. Dieser Algorithmus wertet die Länge des Nutzdatenteils der Datenpakete aus dem Speicher der Maschine aus.
Jedes Paket besteht aus einem Header mit 80 Bits, gefolgt von dem Datenteil mit variabler Länge. Ein Paket beginnt mit dem festgelegten Bitmuster 1110. Der Header enthält eine 2 Bytes lange Kanalnummer, die bei der Bitstelle 32. beginnt. Zu einem Kanal gehören mehrere Datenpakete mit einer eindeutigen Kanalnummer. Die Anzahl der Bits, die in den Speicher geladen werden, wird als bekannt vorausgesetzt und wird in ein entsprechendes Register vorgeladen.
Nun soll der "Paketanalyse"-Algorithmus eine Tabelle, die Kanalnummern und zugehörige Datenlängen (in Bits) enthält, anlegen. Haben mehrere Paket dieselbe Kanalnummer, so werden die Längen des Nutzdatenteils zusammengefügt. Die Tabelle soll ab einer beliebigen Speicheradresse außerhalb des Paketfeldbereichs im Hauptspeicher der Maschine abgelegt werden.
Diese Aufgabenstellung soll mit dem gegebenen Minimax-Simulator simuliert und getestet werden. Die Maschine kann durch vorgegebene Bauteile erweitert werden, was sich jedoch auf die Bewertung auswirkt. Der Algorithmus wird in Form der Steuertabelle implementiert und soll außerdem als Flussdiagramm abgegeben werden.

3 Ist-Analyse der Basismaschine

Die Minimax-Maschine ist ein minimales Rechensystem welches im Wesentlichen aus einigen Registern (Basis ACCU, PC, IR, MDR, MAR) und einer arithmetisch-logischen Einheit (ALU), welche den Datenpfad bilden, und einem Hauptspeicher (HS) besteht. Jeder Register hat als Eingang mindesten die ALU und zusätzlich einen Eingang, der den Schreibzugriff regelt. Der Befehlsablauf des Systems wird über ein Mikroprogramm festgelegt.
Für das Projekt kann die Architektur der Minimax-Maschine um zussätzliche Register bzw. Sign Extension Units erweitert werden.

- Die Basismaschine:

  1. ACCU: Abkürzung für Accumulator ein Zwischenspeicher, um mit MDR Operationen durchführen zu können.
  2. PC: Abkürzung für program counter, enthält den Programmzähler, welcher den nächsten Befehl beeinflusst.
  3. IR: Abkürzung für instruction register, enthält Opcode(8 Bit) und Adressteil(24 bit).
  4. MDR: Abkürzung für memory data register, enthält je nach Einstellung des Multiplexers textttMDR.Sel, verschiede Daten entweder aus Hauptspecher oder aus der ALU.
  5. MAR: Abkürzung für memory adress register, enthält die Speicheradresse an der ausdem Hauptspeicher Daten geladen oder geschrieben werde sollen.
  6. HS: Abkürzung für Hauptspeicher, er wird mit 24-Bit durch MAR adressiert und gibt eine 32-Bit Zahl zurück. Auf die selbe Art funktioniert schreiben einer 32-Bit Zahl natürlich auch.

- ALU-Operationen der Basismaschine:

  1. ADD: Addiert ALU-Eingang A und ALU-Eingang B
  2. SUB.B: Subtrahiert ALU-Eingang A von ALU-Eingang B
  3. TRANS.A: Schaltet den ALU-Eingang A durch
  4. TRANS.B: Schaltet den ALU-Eingang B durch

Die möglichen Operationen auf die in der ALU implementieren Operationen sind beschränkt (Basis: ADD, SUB, TRANS.A, TRANS.B) . Die ALU kann aber mit zusätzliche Operationen , wie z.B dem DIV-Befehl oder dem AND-Befehl, ergänzt werden. Um eine Operation auszuführen müssen über die Multiplexer ALUSel.A und AluSel.B zwei Operanden ausgewählt werden und der ALU muss über die ALU Ctrl-Leitung der Code für die Operation übergeben werden. An den Multiplexern liegen sowohl Konstanten als auch die Register an, welche zur ALU durchgeschaltet werden können. Das Ergebnis der Operation kann entweder in einem Register (über MDR) oder im HS (Adresse im Register MAR) gespeichert werden. Zusätzlich können Flags (Basis nur ein Flag ALU RESULT == 0) gesetzt werden, welche zurück zur Control Unit (CU) geleitet werden um z. B. bedingte Sprünge auszuführen. Die uns vorliegende Minimax-Maschine arbeitet mit 32-Bit und speichert Werte mit 32-Bit in den Registern und im HS. Alle ALU-Operationen werden folglich mit 32-Bit ausgeführt. Dies stellt sich jedoch für unser Aufgabe als Hindernis, da wir die Daten bitweise untersuchen müssen, Daten aus dem HS und den Registern jedoch nur als 32-Bit Zahlen auslesen können und nicht als einzelne Bits.
Folglich wird die Basismaschine um einige Konstanten, Operationen und Register erweitert werden müssen, welche im Implementierungskonzept näher aufgeführt sind.

4 Beschreibung des Implementierungskonzeptes

Bei der Implementierung der Paketanalyse muss zwischen mehreren Teilschritten unterschieden werden. Zunächst wird nach dem Startbitmuster des ersten Paketes gesucht (1). Ist ein Paket identifiziert, können die ersten 32 bits (dies entspricht einer Speicherzelle) übersprungen werden (2). Somit beginnt mit der zweiten Speicherzelle die Kanalnummer des Paketes, diese wird extrahiert und abgespeichert (3). Nach 32 weiteren Bits kommt nun der Datenteil, welcher Bit für Bit ausgelesen wird, während gleichzeitig ein zählregister inkrementiert wird (4). Während dieses Vorgangs wird permanent überprüft, ob ein neues Startbitmuster vorliegt, sodass ein neues Paket anfängt (5). Ist dies der Fall, wird das Zählregister auf den momentanen Wert für die Kanalnummer addiert. Die Kanalnummer dient dabei als Offset für die Speicherzelle (6).

4.1 Identifizierung des Startbitmusters

Da für das Startbitmuster nur die vier niedrigwertigsten Bits untersucht werden, müssen zunächst für einen Vergleich die restlichen ausgeblendet werden. Dies erfolgt durch eine bitwise-and verknüpfung mit Nullen. Anschlieend wird das Ergebnis mit dem Startbitmuster verglichen. Sind sie nicht identisch, wird die Speicherzelle geshiftet. Dies wird solange wiederholt, bis das Ergebnis identisch ist, dann wird mit Schritt 2 fortgefahren.

4.2 Kanalnummer extrahieren

In der ersten Speicherzelle stehen nun die 4 bits des Startmusters, sowie 28 bits die übersprungen werden. Also enthält die erste Speicherzelle keine weiteren Informationen, sodass mit der nächsten Speicherzelle fortgefahren wird. Nun müssen die ersten 16 bits, die die Kanalnummer beinhalten, extrahiert werden. Hierfür sorgt wieder die bitwise-and Verknüpfung der restlichen Bits mit Nullen. Die Kanalnummer muss für die spätere Nutzung abgespeichert werden.

4.3 Datenbits zählen

Auch die zweite Speicherzelle enthält nun keine Informationen mehr. Da in der dritten Speicherzelle noch 16 bits sind, die nicht zum Datenteil gehören, werden im ersten Durchlauf nur 16 auf das Zählregister für die Länge des Datenteils addiert. Für jeden weiteren Schleifendurchlauf werden dann 32 addiert.

4.4 Auf neues Startbitmuster überprüfen

Während die Datenbit nach und nach durch shiften gezählt werden, wird permanent auch der Test aus (1). durchgeführt. Sobald das Startbitmuster erkannt ist, bricht die Schleife aus (3) ab und es folgt Schritt (5).

4.5 Zählregister zur Tabelle hinzufügen

Um die Tabelle zur Speicherung der Längen von den eigentlichen Daten zu trennen, darf diese erst in einer Speicherzelle nach der maximalen Datenlänge beginnen. Die Adresse der ersten Speicherzelle für die Adresse soll dabei

in einem Register gespeichert sein. Nun wird die Kanalnummer auf dieses Register addiert, um die Speicherzelle für eine bestimmte Kanalnummer zu erhalten. Damit hat die Kanalnummer 0 die erste Speicherzelle der Tabelle, die Kanalnummer 1 die zweite, usw. . Auf den Inhalt dieser Speicherzelle wird dann der Inhalt des Zählregisters addiert. Anschließend werden die temporären Register zurückgesetzt und der Algorithmus beginnt erneut mit Schritt (2).

5 Angestrebte Projektergebnisse

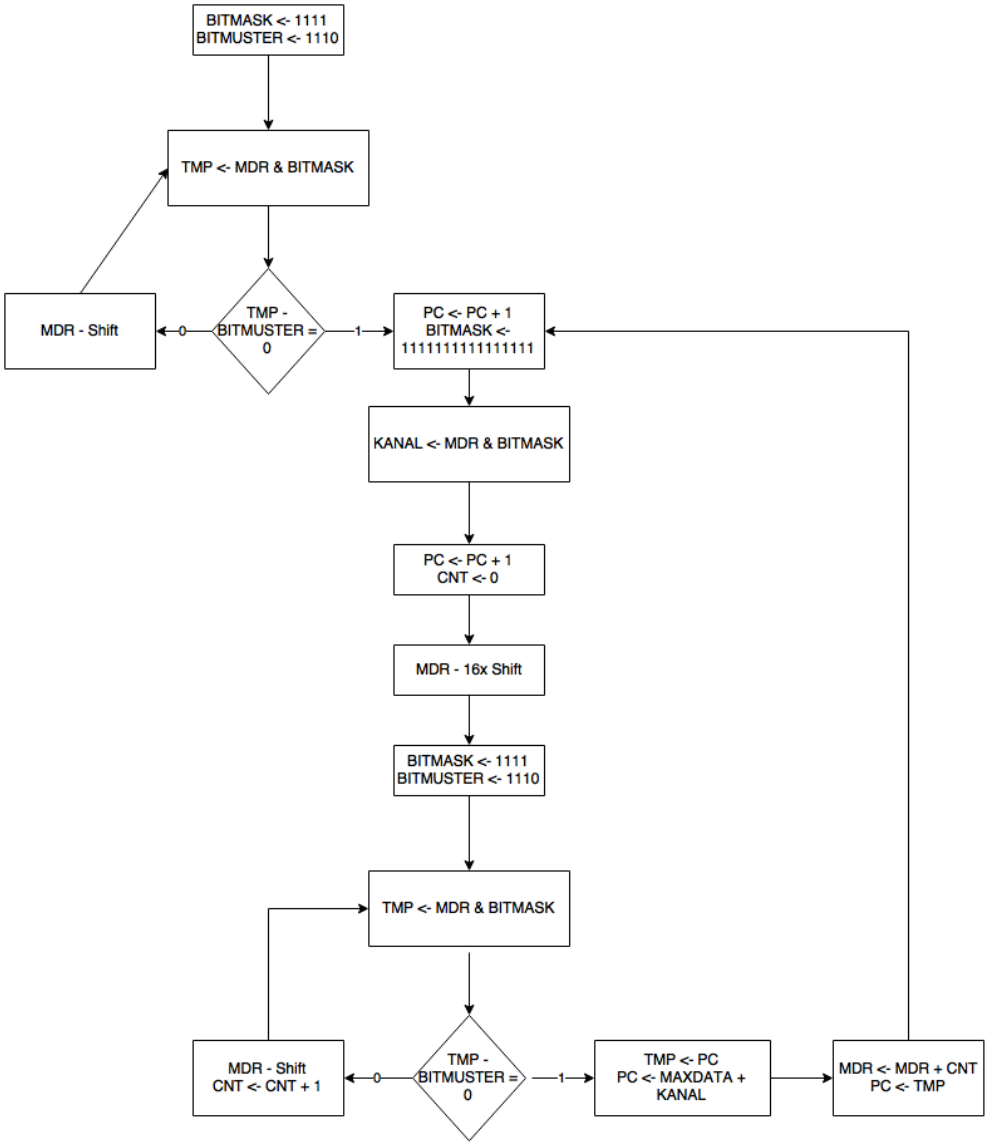
- Ein Pflichtenheft, in dem wir unsere Vorüberlegungen festhalten.
- Eine vollständige und funktionsfähige Lösung mit allen Testdateien.
- Einen Schaltplan, bei dem wir die gegebene Minimax - Maschine um die notwendigen Elemente erweitert haben.
- Eine vollständige Dokumentation, mit der wir unseren Algortihmus und unsere implementierte Lösung beschreiben und erläutern.

6 Arbeitsaufteilung

- Maximilian Lumpe: Einleitung und Aufgabenstellung
- Phu Bac Duoung: Ist-Analyse der Basismaschine
- Jan Feuchter: Beschreibung des Implementierungskonzeptes
- Niklas Blume: Angestrebte Projektergebnisse und Anhang

7 Anhang: Flussdiagramme geplanter Mikroprogramme

7.1 Kompletter Algorithmus



8 Anhang: Verwendete Hilfsmittel

- Vorlesung: Grundlagen der Rechnerarchitektur
- Umdruck Paketanalyse 2016/17
- Minimax - Simulator
- GitHub zur Versionskontrolle und Organisation
- [https://www.draw.io] zur Erstellung der Flussdiagramme