Bau eines Platinen Computers

Alexander Wersching und Simon Walter

2021

Inhaltsverzeichnis

1 Anforderungen

2 Specification des Computers

Eine Computer hat eine Reihe Verschiedener high-level Aufgaben und Features die es ihm erlauben seiner Tätlichkeiten nach zu gehen! Diese Features können wie folgt auf gegliedert werden: Register, Memory, Banking, Input/Output, Flagen, Hardware Stack, Interrupts, Supervisor-Mode und das Instructuion Set.

Aber um zuerst ein paar standarts festzulegen: der Computer hatte eine Daten/Addresse Länge von 16 bit. D.h. er kann auf maximal 16 bit aufeinmal operieren bzw können maximal 16 bit an Daten/Addresse im Computer auf einmal bewegt werden.

2.1 Registers

Register sind kleine sehr schnelle Speichereinheiten, die genutzt werden können um zwischen Ergebnisse, local Variablen, argumente für Funktionen zu speichern oder um Daten im Computer zwischen den einzelnen Komponenten zuverschieben. Aus jedem Register kann *gelesen* werden (*Kopiert* des alten Wert an eine anderen Ort) oder *geschrieben* werden (den alten Wert mit einem neuen Wert *überschrieben*)

Dem Programmiere des Computers stehen bei uns 8 verschieden, jeweils 16 bit lange Register zur verfügung. Jedes Register hat dabei eine eigene Register Number (= RN). Machen Register erfühlen zudem noch speziell Aufgaben und sollten auch nur so verwendet werden. Diese Funktionen sind in Tabelle 1 zu sehen.

Tabelle 1: Register Aufteilung

RN	Name	Funktion
0	ZERO	Konstante 0x0000; Schreiben hat keine Effekt
1	IP	Instruction Pointer, zeigt auf akutuelle auszuführenden Instruktion
2	SP	Stack Pointer, zeigt auf obersten Element des Hardware Stacks, siehe 2.3.2
3	A	Alzweck Register ohne besonderen Aufgaben
4	B	...
5	C	...
6	D	...
7	FLAG	Enthält alle wichtigen Flaggen, siehe 2.1.1

Die Register A-B sind dabei die Register, dabei freie für jede nutzung. Es ist *stark zuempfelen* die spezieall Register, (ZERO, IP, SP und FLAG) *nicht* für die speicherung von Daten zuverwenden, da diese evt. Program abläufe stark störenen können. Sie sind ausdrücklich nur für ihre zugewiesenen Aufgaben da.

2.1.1 Flaggen

Das Register FLAG, enthält alle Status Flagen. Ein Flagge kann entweder gesetzt oder nicht gesetzt sein, daher kann jede Flagge als ein bit repräsentiert werden. Jedes Bit in FLAG stellt daher eine Flagge da. Flaggen werden bei unterschiedlichen Ereignissen geschaltet. Dies, und welche Flaggen es gibt ist in Tabelle 2

Tabelle 2: Flaggen auf Teilung in FLAG

Bit	Name	Funktion	Schaltung
0	OF	Intager Overflow einer Rechenoperation an	Bei jeder ALU Operation
1	E	Gleichheit zweier Werte an	TEST arg[0] = arg[1]
2	G	Ungleicheit (>) zweier Werte an	TEST arg[0] > arg[1]
3	S	Ungleicheit (<) zweier Werte an	TEST arg[0] < arg[0]
4	P	IO-Geräte Eingesteckt siehe 2.4	TEST arg[0]
5	SOP	Supervisor-/Usermode	Interrupt siehe 2.5
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

Alle Flagen die mit TEST als Schaltung gekenzeichnet werden, werden bei der Ausführung der TEST Operation geschaltet. Da die TEST Operation

Das FLAG Register verhält sich ebenfalls wie ein Register, daher können Flagge auch per Hand gesetzt oder nicht gesetzt werden. Eine Ausnahme macht SOP die nur manuel geschaltet werden kann wenn SOP gesetzt ist.

2.2 Supervisor/Normal Mode

2.3 Memory

2.3.1 Banking

2.3.2 Hardware Stack

2.4 Input/Output

Der Computer hat zudem die Möglichkeit mit verschieden IO-Geräte zu interagieren. Diese läuft über die sogenaten IO-Ports. Der Computer hat physische Slots wo IO-Geräte angesteckt werden können. Dabei bekommt jedes IO-Geräte eine eigene ID (IO-ID), welche im Bereich von 3 bis 65535 liegt. Die ersten drei IO-IDs sind für spezielle Aufgaben zugewiesenen. Die Interaktion funktioniert dabei wie mit Memory. Es kann zudem IO-Geräte Daten übertragen werden mit der OUT Instruktion und Daten vom IO-Gerät gelesen werden über IN Instruktion. Die IO-ID sind dann wie folgt vergeben, wie Tabelle 3

Tabelle 3:

IO-ID	Funktion
0x0000	IO-Gerätetyp
0x0001	Interrupt Quelle
0x0002	Interrupt ID
...	
0xffff	physische IO-Geräte

Jedes IO-Geräte hat eine feste unveränderliche Typen nummer. Diese lassen sich über die IO-ID 0x0000 herausfinden. Diese sieht dann wie folgt aus:

```
OUT A 0x0000 ; A enthaelt die IO-ID fuer ein unbekanntes Geraet
IN  B 0x0000 ; B enthaelt nun die Typennummer des Geraets
```

2.5 Interrupts

2.6 Instructuion Set

3 Kommentar zur Specification

3.1 16 bit Daten/Address Länge

Die Address- und Datenlänge besagt viele Binäre Ziffern (bits) für eine Address und Daten verwendet werden. Je mehr Ziffern man verwendet desto größere Zahlen können in einer Operation verarbeitet werden wobei die maximal repräsentierbare Zahle $2^n - 1$ ist wenn n die Anzahl der Binary Ziffern (= Bits) ist und wenn man mit 0 anfängt zu zählen. Sie Tabelle 4

Ein eine n stellige Binär Zahl wobei n teilbar durch 8 ist, ist dabei relative angenehm, 8 bit = 1 byte. Das byte ist dabei die Basis Einheit für so gut wie alle informationstechnischen Standards. Daher ist es gute eine solches n zu wählen. Zudem kommen einige elementar ICs, z.B. Buffer oder Register, immer mit 8 bits.

Jedoch gilt je mehr Bits man verwendet, desto mehr Schaltung brauchen man, desto mehr Stromverbrauch. Da wir wie zwar nicht speziell auf Platz und Stromverbrauch optimieren, aber trozdem ein limitiertes Buget haben, können wir auch nicht eine belibig große Daten/Address Länge wählen.

Wir haben uns daher für 16 bit entschieden, da man laut Tabelle 4 Zahlen bis zwischen 0 und 65535 darstellen kann. Diese Menge ist gerade große genug, um für die meisten Programme große genüge Zahl abzubilden [1].

Die Entscheidung zur größe der Daten/Address Länge ist eine sehr ausschlag gebende, daher haben wir diese zuerst festgelegt. Sie wird für viele weitere Entscheidung eine wichtige Rolle spielen. Wichtig ist nur das wir maximal 65536 Zustände oder Zahlen von 0 bis 65535 darstellen können mit unseren 16 bit

[1] Für Programme die Zahlen welche > 65535 darstellen wohlen ist es ratsam immer zwei oder mehr Spreichereinheiten zunehmen oder bei Berechnungen, selbige auf zwei oder mehr Schritte auf zuteilen

Tabelle 4: Maximalerepersäntierbare Zahlen (startend mit 0)

n	$n^2 - 1$
2	3
4	15
8	255
16	65535
32	4294967295
64	18446744073709551615

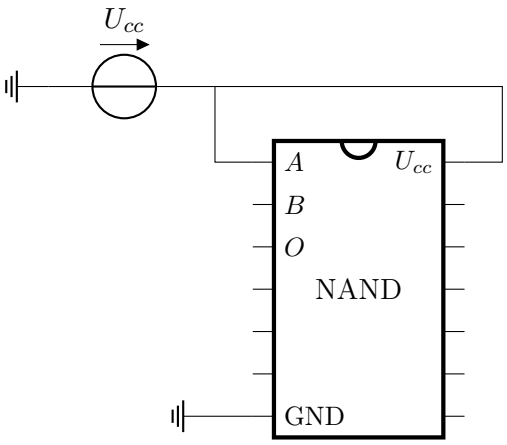
3.2 Register Anzahl

4 Grundlegen Tests der Einzelnen Komponenten

4.1 NAND vs AND,OR,NOT

4.2 NANDs

Das NAND Gatter ist ein fundermentaller Logik Bautstein und bedarf daher einer Näheren Inspektion. Jedes NAND Gatte hat einen bestimmten propagation delay. Dieser gliedert sich einmal auf in High to Low (t_{PHL}) und Low to High (t_{PLH}). Diese ist den Datenblättern zu entnehmen. Wir haben uns dazu entschieden diese noch einemal selbst zutesten. Wir haben zu dem unterschiedliche NANDs von Unterschiedlichen Herstellern getested. Der Test Aufbau war dabei immer gleich:



Wir haben alle Test mir einer Basis Spannung von $U_{cc} = 5V$, $V_H = 5V$ und $V_L \leq 0.7V$ durch deführt

4.3 SRAM

Der RAM oder Random Access Memory ist der Hauptspeicher des Computers und bedarf daher ebenfalls einer genaueren Untesuchen. Er hat viele Funtionen die Getestet werden müssen:

1. Zeit zwischen OE und dem Erscheinen der Daten

2. Zeit zwischen Änderung der Addresse und dem Erscheinen der Daten

4.4 ROM

4.5 Buffer

Der Test Aufbau ist dabei wie folgt:

5 Quellenverzeichnis