Technische Informatik

bei Prof. Dr. Joachim Klaus Anlauf

18.02.2020

Es gab $x + x + x + x = 72$ Punkte insgesamt, die Bearbeitungszeit betrug 2 Stunden. Die Aufgabenstellung wurde im Wortlaut wiedergegeben und kann leichte Abweichungen beinhalten sowie unvollständig sein.

1. (6 Punkte) **Quickies**
   Es gibt 0.5 Punkte Abzug für jede falsch oder garnicht beantwortete Frage. Dabei können Teilaufgaben minimal 0 Punkte geben.

   (a) (1 Punkt) Operatorensysteme:
   - ☐ ☐ Die drei Werte der Schaltalgebra sind 0, 1, -1
   - ☐ ☐ Die beiden Operationen in der Schaltalgebra sind Durchschnitt und Vereinigung
   - ☐ ☐ Schaltalgebra ist ein Spezialfall der Boolsche Algebra

   (b) (1 Punkt) Multiplexer:
   - ☐ ☐ Ein Multiplexer hat 2^n Eingänge und einen Ausgang bei n Steuerleitungen
   - ☐ ☐ Ein Demultiplexer hat 1 Eingang und 2^n Ausgänge bei n Steuerleitungen
   - ☐ ☐ Ein Demultiplexer hat 2^n Eingänge und einen Ausgang bei n Steuerleitungen
   - ☐ ☐ Ein Adressdekoder hat n Eingänge und 2^n Ausgänge

   (c) (1 Punkt) Addierer:
   - ☐ ☐ Ein CLAA ist schneller als ein Ripple Carry Addierer
   - ☐ ☐ CLAA Berechnet berechnet die Summe von 2 n-Bit Zahlen
   - ☐ ☐ Ein CLAA berechnet die Summe zweier Zahlen in $\mathcal{O}(1)$
   - ☐ ☐ Ein CLAA macht aus 3 Zahlen 2 Zahlen mit derselben Summe

   (d) (1 Punkt) Pipelines:
   - ☐ ☐ Eine Pipeline versorgt Hochleistungprozessoren mit Kühlwasser
   - ☐ ☐ Der Sinn eine Pipeline ist es, den Befehlsdurchsatz zu erhöhen
   - ☐ ☐ Pipelining verringert die Latenzzeit eines Befehls
   - ☐ ☐ Die Anzahl der Pipelinestufen ist proportional zum Beschleunigungsfaktor der Pipeline

   (e) (1 Punkt) Mikroprogramm:
   - ☐ ☐ Ein Mikroprogramm heißt so, weil man es nur unter dem Mikroskop sehen kann
   - ☐ ☐ Ein Mikroprogramm besteht aus Befehlen wie „ADD #5“
   - ☐ ☐ Mit einem Mikroprogramm kann man Steuersignale für einen Datenpfad erzeugen
   - ☐ ☐ Ein Mikroprogrammbefehl besteht aus drei Teilen: Daten, Bedingungen und Mikrobefehlsadressen

   (f) (1 Punkt) Speicherzellen
   - ☐ ☐ SRAM flüchtig
   - ☐ ☐ DRAM flüchtig
   - ☐ ☐ Flash ist ein Spezialfall von EEPROM
   - ☐ ☐ Nibble-Mode ist langsamer als Page-Mode

2. (8 Punkte) **Zahlensysteme**
   (a) (3 Punkte) Entwickeln Sie ein Zahlensystem zur Basis 7. Rechnen Sie die Dezimalzahl 78 in ihr System um.
   (b) (5 Punkte) Gegeben seien die Zahlen C0C00000 und C0200000 im IEEE 754 Single Precision Format. Addieren Sie die zwei Zahlen und geben Sie das Ergebnis in IEEE 754 Single Precision Format als Hexadezimalzahlen an.

3. (6 Punkte) **Boolsche Alge*bruh***

   (a) (2 Punkte) Zeigen oder widerlegen Sie, dass die Operatorenmenge $\{\leftrightarrow, \wedge, 0\}$ ein vollständiges Operatorensystem bildet:

   (b) (2 Punkte) Gegeben sei die folgende boolsche Funktion:

   $$((a \leftrightarrow b) \leftrightarrow \neg b) \rightarrow a$$

   Wenden Sie die Shannonzerlegung für die Variable a an und vereinfachen Sie so weit es geht. Geben Sie bei jedem Schritt die verwendete Regel an.

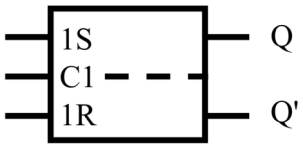
4. (9 Punkte) **Quine-McCluskey**
   Wenden Sie das Quine-McCluskey-Verfahren an, um die KNF der Funktion herauszufinden.

a	b	c	d	f
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	-
0	1	0	1	1
0	1	1	0	0
0	1	1	1	-
1	0	0	0	-
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

5. (6 Punkte) Erstellen Sie aus der gegebenen Schaltung einen Komparator. Verwenden eine minimale Anzahl von NOT, OR, AND, NOR, NAND, und XOR-Gattern. Die Zahlen a und b liegen im Zweierkomplement vor.

6. (6 Punkte)

   (a) (2 Punkte) Erstellen Sie aus dem gegebenen R-S-Flipflop ein D-Flipflop. Dafür können Sie Standardgatter verwenden.



   (b) (4 Punkte) Zeichnen Sie einen synchronen Zähler unter der Verwendung von ausschließlich D-Flipflops, der die folgende Belegung ergibt:

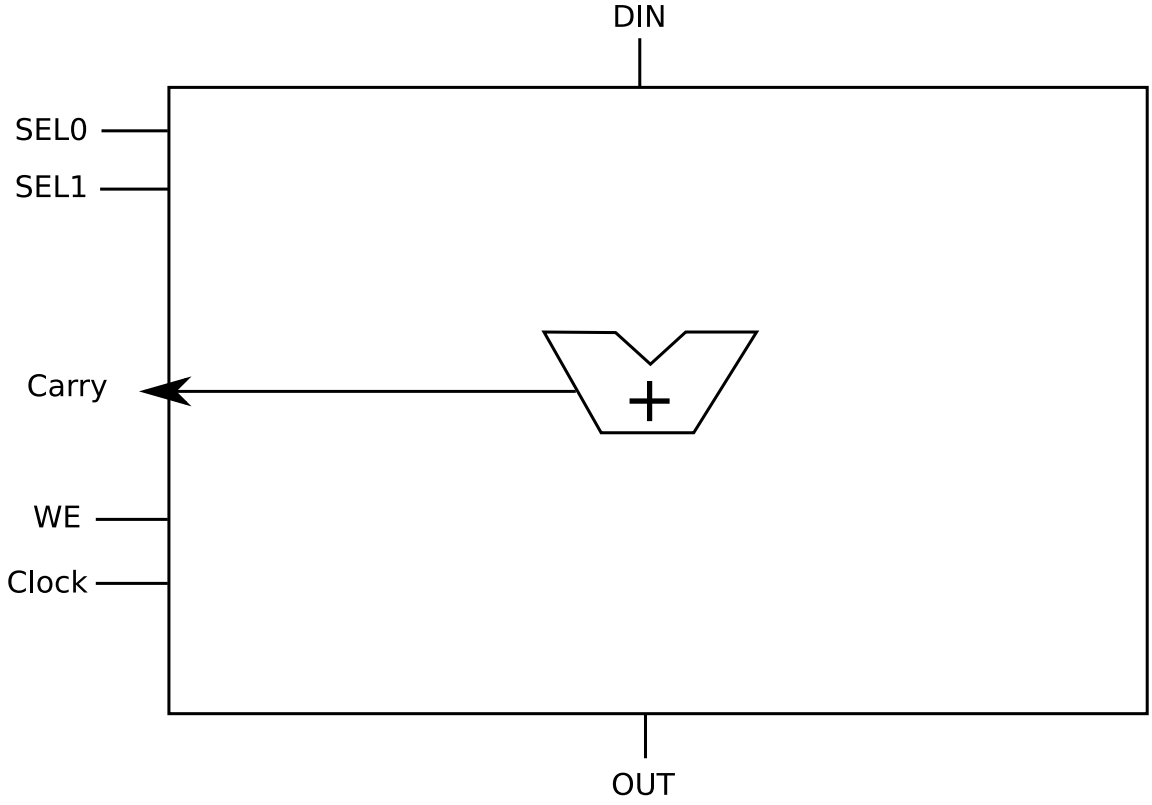
q_1	q_0	q_1^+	q_0^+
0	0	1	0
1	0	1	1
1	1	0	1
0	1	0	0

7. (13 Punkte)

   (a) (5 Punkte) Der Datenpfad soll $\sum_{i=1}^{3} a_i$ berechnen. Implementieren Sie eine Schaltung, die folgende Register-Transfer-Operationen realisiert:

   $R \leftarrow 0$
   $R \leftarrow$ DIN
   $R \leftarrow$ R+DIN

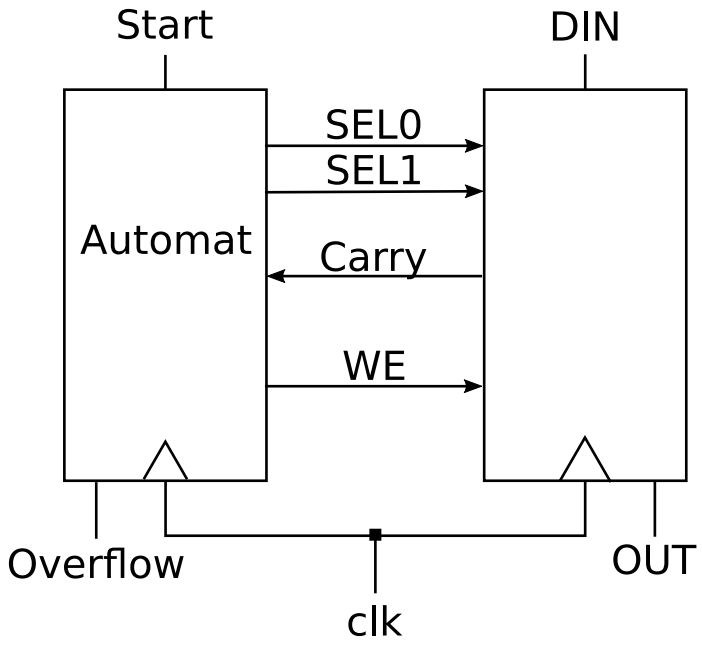
   WE steht für write-enable, SEL0 und SEL1 sind 1 Bit breite Steuerpfade.



(b) (2 Punkte) Geben Sie die Belegungen der Leitungen für die Register-Transfer-Operationen an.

SEL0	SEL1	WE	Operation
			$R \leftarrow 0$
			$R \leftarrow$ DIN
			$R \leftarrow$ R+DIN

(c) (6 Punkte) START ist genau dann 1, wenn am Eingang DIN a_1 anliegt. Am Anfang ist OVERFLOW 0, am Ende der Rechnung liegt bei OUT das Ergebnis der Rechnung
Start=1 $\iff$ $a_1 = 1$, Addieren. Wenn Carry=1: 1 Takt Overflow=1, OUT=0, dann auf Start warten. Zeichnen Sie einen Mealy-Automaten, der das obige Schaltwerk steuert.



8. (10 Punkte) **Assembler**

Nr	Befehl	Codierung				Beschreibung
0	NOP	0	0	0	0	Wartezyklus (*No Operation*)
						Lade- und Speicherbefehle
1	LDA #n	0	0	0	1	Lädt den Akkumulator mit dem Wert *n*
2	LDA (n)	0	0	1	0	Lädt den Akkumulator mit dem Inhalt der Speicherstelle *n*
3	STA n	0	0	1	1	Überträgt den Akkumulatorinhalt in die Speicherstelle *n*
						Arithmetikbefehle
4	ADD #n	0	1	0	0	Erhöht den Akkumulatorinhalt um den Wert *n*
5	ADD (n)	0	1	0	1	Erhöht den Akkumulatorinhalt um den Inhalt der Speicherstelle *n*
6	SUB #n	0	1	1	0	Erniedrigt den Akkumulatorinhalt um den Wert *n*
7	SUB (n)	0	1	1	1	Erniedrigt den Akkumulatorinhalt um den Inhalt der Speicherstelle *n*
						Sprungbefehle
8	JMP n	1	0	0	0	Lädt den Instruktionszähler mit dem Wert *n*
9	BRZ #n	1	0	0	1	Addiert *n* auf den Instruktionszähler, falls das Zero-Bit gesetzt ist
10	BRC #n	1	0	1	0	Addiert *n* auf den Instruktionszähler, falls das Carry-Bit gesetzt ist
12	BRN #n	1	0	1	1	Addiert *n* auf den Instruktionszähler, falls das Negations-Bit gesetzt ist

Schreiben Sie unter Verwendung des obigen Befehlssatzes ein Assemblerprogramm, das ermittelt, ob a durch b teilbar ist (a,b positive Ganzzahlen). Gehen Sie davon aus, dass sich a und b an den Speicheradressen 13 und 14 befinden. Wenn b Teiler von a ist, soll nach der Beendigung des Programms an der Speicheradresse 15 eine 1 vorliegen, wenn b kein Teiler von a ist eine 0. Kommentieren Sie **jede** Zeile Code!

Zeile	Label	Code	Kommentar
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

9. (8 Punkte) **Cache**

   (a) (2 Punkte) Ein 16-Bit Computer greift nacheinander auf folgende Adressen zu (dezimal):

   0, 3, 8, 4, 7, 4, 12, 7, 4, 3

   Jeder Zugriff auf den Hauptspeicher benötige 5 Takte. Wie viele Takte werden für obige Speicherzugriffe benötigt?

   (b) (6 Punkte) Nun soll der Computer einen kleinen Direct-Mapped Cache mit 4 Cachezeilen bekommen. Bei jedem Hit benötigt der Speicherzugriff nur noch einen Takt, bei einem Miss sind es immer noch 5 Takte. Tragen Sie den Zustand des Caches nach Abarbeitung der Speicherzugriffe ein. Tragen Sie ebenfalls die Hauptspeicheradressen ein (in einem richtigen Cache stünden hier die Daten). Streichen Sie bei Bedarf *(immer, sonst Punktabzug!)* überschriebene Tags und Adressen durch. Tragen Sie gleichzeitig die Adressen in der Reihenfolge ihres Auftretens in die zweite Tabelle ein und vermerken Sie bei jedem Zugriff, ob es sich um einen Hit oder einen Miss handelt.

Zeile	Valid	Tag	Hauptspeicheradresse
0			
1			
2			
3			

Hauptspeicheradresse	Hit

   (c) (1 Punkt) Wie viele Takte benötigt der Computer nun bei derselben Abfolge von Adressen?