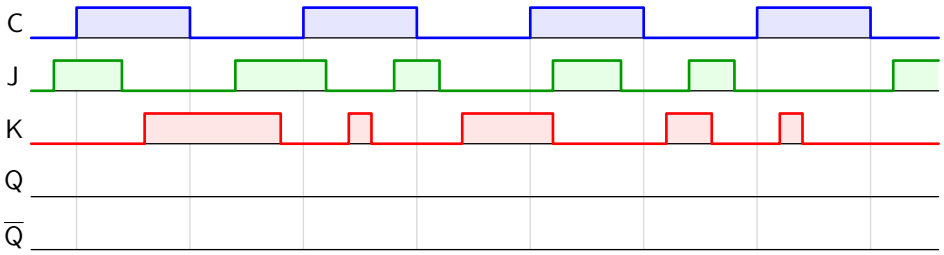
Rechnersysteme und -netze Winter 2017/2018

Christian Borgelt, Christoph Doell,
Heiko Dreyer, Alice Hildebrand, Josua Sattler, Simon Suckut

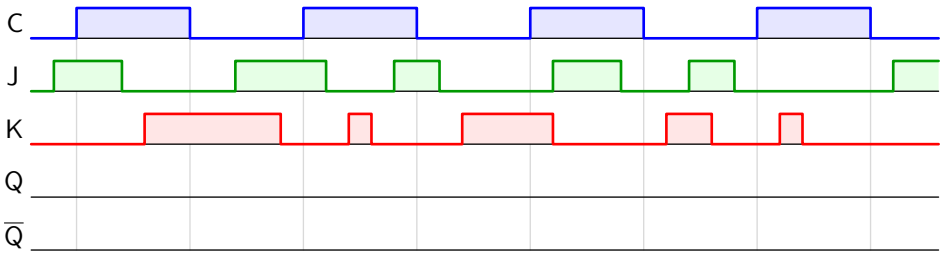
5. Übungsblatt

Aufgabe 1 Riegel und Flipflops

a) Bestimmen Sie für die unten gezeigten Eingaben C, J und K die Ausgaben Q und $\overline{\mathsf{Q}}$ für einen JK-**Riegel**! Zeichnen Sie diese in das Diagramm!



b) Bestimmen Sie für die unten gezeigten Eingaben C, J und K die Ausgaben Q und $\overline{\mathsf{Q}}$ für ein JK-**Flipflop**! Zeichnen Sie diese in das Diagramm!



Aufgabe 2 Arithmetik

a) Konvertieren Sie die Zahl 1101100111110110_2 ins Hexadezimalsystem und die Zahl $4AC9E_{16}$ ins Binärsystem!

b) Gegeben seien die beiden vorzeichenlos zu interpretierenden Binärzahlen $a = 00101_2$ und $b = 11101_2$. Berechnen Sie $a + b$ und $a - b$ im Binärsystem! Überprüfen Sie Ihre Rechnung im Dezimalsystem!

c) Gegeben seien die beiden im Zweierkomplement zu interpretierenden 4-Bit-Zahlen $x = 1011_2$ und $y = 0111_2$. In der Vorlesung wurde das Produkt $y \cdot x$ berechnet. Berechnen Sie nun mit dem gleichen Schema (**ohne** Ausnutzen der Kommutativität) $x \cdot y$! Vergleichen Sie Ihr Ergebnis mit dem der Vorlesung!

Aufgabe 3 Arithmetisch-logische Einheit

In der Vorlesung wurde die arithmetisch-logische Einheit der Hack-Architektur aus „The Elements of Computing Systems: Building a Modern Computer from First Principles“ betrachtet. Die von ihr zu berechnende Funktion wird durch sechs Steuerbits bestimmt: zx, nx, zy, ny, f und no. Die obere Hälfte der folgenden Tabelle wurde in der

Vorlesung erklärt. Erklären Sie nun für die untere Hälfte (analog zur Vorlesung) wie die Steuerbits die entsprechenden Outputs verursachen.

These bits instruct how to preset the input x		These bits instruct how to preset the input y		This bit selects between + and &	This bit instructs how to postset the output out	Resulting ALU output
zx	nx	zy	ny	f	no	out=
if zx then x = 0	if nx then x = ~x	if zy then y = 0	if ny then y = ~y	if f then out = x + y else out = x & y	if no then out = ~out	f(x,y) =
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	-1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	~x
1	1	0	0	0	1	~y
0	0	1	1	1	1	-x
1	1	0	0	1	1	-y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x-1
1	1	0	0	1	0	y-1
0	0	0	0	1	0	x+y
0	1	0	0	1	1	x-y
0	0	0	1	1	1	y-x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y

Legen Sie besonderen Wert auf die Erklärung der Subtraktion! Denn man beachte: Diese arithmetisch-logische Einheit implementiert offenbar die Subtraktion *nicht*, wie an früherer Stelle in der Vorlesung gezeigt, durch eine Kette von XOR-Gattern (zur bedingten Berechnung des Einerkomplementes) und die Eingabe eines Übertrags (*carry*) für die unterste Binärstelle (zur Addition von 1, um das Zweierkomplement zu erhalten), sondern auf andere Weise. Was ist das Prinzip dieser Berechnung?
Beachten Sie, dass die Vorlesungsfolien zu diesem Thema kürzlich aktualisiert wurden, um mögliche Mehrdeutigkeiten zu erläutern!

Zusatzaufgabe Implementierung der Multiplikation

Die arithmetisch-logische Einheit (ALU) des Hack-Systems ist sehr einfach und folglich in ihren Funktionen sehr eingeschränkt. So kann sie z.B. keine allgemeine Multiplikation von Binärzahlen, ja nicht einmal eine Multiplikation mit einer Zweierpotenz direkt ausführen (denn es gibt z.B. keine Bit-Schiebe-Operation). Wie kann man mit den Möglichkeiten dieser arithmetisch-logischen Einheit dennoch ein Analogon des Standard-Multiplikationsalgorithmus implementieren?

Als Antwort auf diese Frage ist kein konkretes Programm in der Hack-Maschinensprache gesucht, sondern vielmehr ein allgemeines Algorithmenschema, das mit den Operationen der Hack-ALU auskommt, um eine Multiplikation durchzuführen.

Hinweise: Braucht man zur Implementierung des Standardalgorithmus zwingend eine Operation ASR (*arithmetic shift right*, arithmetisches Bit-Schieben nach rechts)? Könnte man stattdessen auch mit einer Bit-Schiebe-Operation nach links arbeiten? Wodurch kann man eine Bit-Schiebe-Operation nach links ersetzen, wenn sie nicht direkt verfügbar ist (wie hier der Fall)? Wie geht man bei diesem (abgeänderten) Algorithmus am besten mit negativen Zahlen um?