Virtuelle Lehrerfortbildung Informatik Niedersachsen

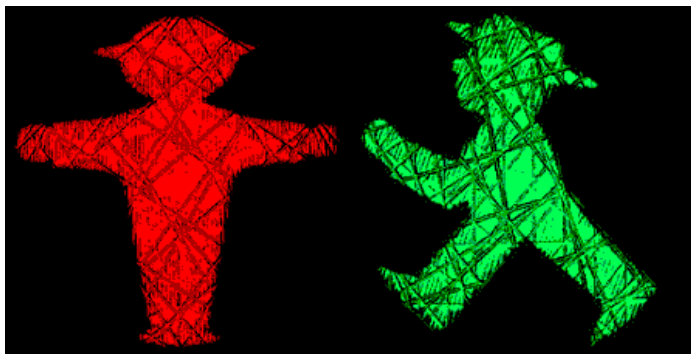
Hans Peter Schneider

Ampelschaltung

Logik löten

Eine virtuelle Unterrichtsreihe im Rahmen des jahrgangsübergreifenden Informatikunterrichts der Oberstufe an Gymnasien

3. Semesterarbeit
zur Virtuellen Lehrerweiterbildung Informatik in Niedersachsen



GYMNASIUM LÜNEBURGER HEIDE

Inhalt

Vorbereitungen 3

Einordnung 3
Unterrichtsinhalte 3
Hilfsmittel 4

Die Unterrichtsreihe 5

Aufgabenstellung 5
Der Automat 6
Die Simulation 7
Die Schaltung 7
Das Modell 9

Quellen 10

Vorbereitungen

Einordnung

Das Informatikprofil gehört am Gymnasium Lüneburger Heide zu den Aushängeschildern. Nach dem für alle Schüler der 6. Klassen verplichtenden PC-Führerschein, in dem den Schülern der Umgang mit dem Rechner, dem Schulnetzwerk, dem Internet und den üblichen Office-Anwendungen beigebracht wird, haben die Schüler die Möglichkeit, Informatik ab der 7. Klasse durchgehend bis zum Abitur als Unterrichts- und Prüfungsfach zu belegen. In der Mittelstufe wird Informatik im Rahmen des Profilunterrichts Klasse 7 bis 9 (bisher Wahlpflichtunterricht der Klassen 9 und 10) angeboten, in der Oberstufe als jahrgangsübergreifender Kurs, zur Zeit als Grundkurs P3/P4 für die Klassen 11 bis 13, zukünftig als Prüfungskurs mit normalen Anforderungen als Ersatz für eine zweite Naturwissenschaft. Die Abiturthemen werden in einem dreijährigen Zyklus wechselnd unterrichtet. Für die Teilnahme am Oberstufenkurs wird die Teilnahme an zwei Mittelstufenkursen vorausgesetzt. Schüler, die Informatik in Klasse 11 neu beginnen, nehmen am Unterricht der Klasse 9 teil. Der Grund für diese Einschränkung ist der, dass in der Mittelstufe drei Halbjahre JAVA-Programmierung unterrichtet wird und diese JAVA-Kenntnisse für den Oberstufenkurs notwendig für eine in ihren Leistungen einigermaßen homogene Schülerstruktur notwendig sind.

Der Aufbau und die Inhalte der Kurse im Einzelnen:

Neue Richtlinien: Zentralabitur und Profilunterricht	Alte Richtlinien: Schulabitur und Wahlpflichtunterricht
Klasse 7 Profilunterricht: 3 Wochenstunden ✍ Tabellenkalkulation, Excel-Programmierung ✍ HTML und CSS, Webseitengestaltung ✍ Einführung in JavaScript	
Klasse 8 Profilunterricht: 4 Wochenstunden ✍ Sequentielle Programmierung mit JavaKara ✍ Kontrollstrukturen von JAVA ✍ Vertiefung in JavaScript und HTML-DOM	Klasse 9 Wahlpflichtunterricht: 4 Wochenstunden ✍ HTML und CSS, Webseitengestaltung ✍ JAVA-Kontrollstrukturen mit JavaKara ✍ Einführung in JavaScript, HTML-DOM
Klasse 9 Profilunterricht: 4 Wochenstunden ✍ Objektorientierte Programmierung mit JAVA	Klasse 10 Profilunterricht: 4 Wochenstunden ✍ Objektorientierte Programmierung mit JAVA
Klasse 10 bis 12 NA-Kurs: 2+4+4 Wochenstunden	Klasse 11 bis 13 Grundkurs: 3+3+3 Wochenstunden
✍ Datenbanken, ER-Modelle und SQL ✍ Informatik und Gesellschaft: Datenschutz, Urheberrecht, soziale Auswirkungen, Geschichte	
✍ Algorithmen und Datenstrukturen I: Suchen und Sortieren auf Feldern ✍ Algorithmen und Datenstrukturen II: dynamische Listen und Bäume, Stack und Pipe	
✍ Endliche Automaten und Parserbau: Zustandsgraphen, Touring-Maschinen, Parser und Interpreter ✍ Technische Informatik: Logik, Rechnerstrukturen	

Unterrichtsinhalte

Die erste Berührung mit Logik hatten die Schüler bereits in der fünften Klasse im Physikunterricht, als sie beim Thema Elektrik einfache Schaltungen aus Experimentierkästen zusammensteckten. In dem Zusammenhang haben sie eine Sicherheitsschaltung (UND), eine Türklingelschaltung (ODER) und eine Flurlichtschaltung (EXKLUSIV-ODER) gebaut.

Das zweite Mal hatten die Schüler Logik im Informatikunterricht der neunten Klasse, als sie sich im Rahmen von JAVA-Kontrollstrukturen mit dem primitiven Typ Boolean und logischen Operatoren

auseinandersetzen mussten. Behandelt wurden in diesem Zusammenhang die in JAVA gebräuchlichen Operatoren **&** (UND), **|** (ODER), **!** (NICHT), **^**bzw. **!=** (EXKLUSIV-ODER, UNGLEICH) und **==** (GLEICH). Nur die ersten drei Operatoren sind elementar, die anderen lassen sich durch die ersten darstellen:

A **^** B identisch mit A **!=** B identisch mit (A **&** **!**B) **|** (**!**A **&** B)

A **==** B identisch mit **!**(A **^** B) identisch mit (A **&** B) **|** (**!**A **&** **!**B)

A	B	!(A & B)
f	f	w
w	f	w
f	w	w
w	w	f

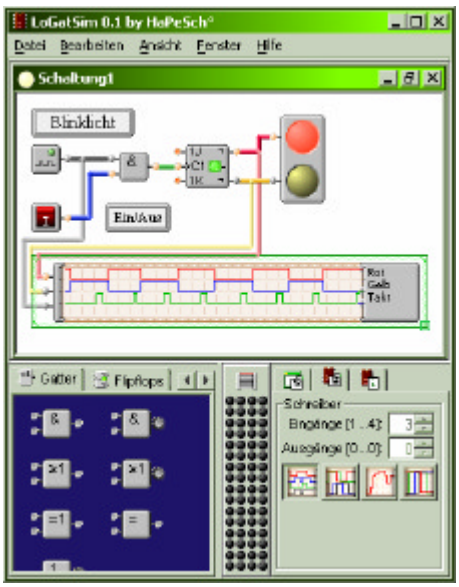
Die Schüler haben logische Tabellen wie die rechts ausgefüllt und die Operatoren bei der JAVA-Programmierung angewendet.

Im Oberstufenkurs wurde vor diesem Unterrichtsprojekt die Schüler in die Themen „Endliche Automaten“ und „Schaltalgebra“ eingeführt. Speziell können die Schüler Zustandsgraphen und -tabellen erstellen und haben einfache technische Schaltungen in einem Simulator gebaut. Sie kennen die einfachen logischen Gatter, die die logischen Operatoren umsetzen, sowie Schalter, Taster, Flipflops, Delay-Bausteine, einen Oszillator, Frequenzteiler und verschiedene Anzeigemöglichkeiten und ihre Einsatzmöglichkeiten aus dem Simulator.

Hilfsmittel

Um die Berechnungen der Ausgangswerte einer Zustandstabelle zu verifizieren, eignet sich besonders gut das Tabellenkalkulationsprogramm eines üblichen Office-Pakets. Wir verwenden hier MS-Excel. Glauben die Schüler, die korrekte Berechnung eines Ausgangs gefunden zu haben, setzen sie diese Berechnung in die Zellen der Tabelle ein und überprüfen, ob das Ergebnis mit dem gewünschten überein stimmt. Excel interpretiert von sich aus den Wert 0 als FALSCH und jeden Wert größer 0 als WAHR. Über die WENN-Funktion lässt sich das Ergebnis einer Operation wieder als 1 oder 0 darstellen. Über die Funktion „bedingte Formatierung“ lassen sich die Ergebnisse auch automatisch hervorheben.

	A	B	C	D
1	S	E	S	A
2	0	0	=WENN(UND(A2;B2);1;0)	=WENN(ODER(A2;B2);1;0)
3	0	1	=WENN(UND(A3;B3);1;0)	=WENN(ODER(A2;B3);1;0)
4	1	0	=WENN(UND(A4;B4);1;0)	=WENN(ODER(A4;B4);1;0)
5	1	1	=WENN(UND(A5;B5);1;0)	=WENN(ODER(A5;B5);1;0)



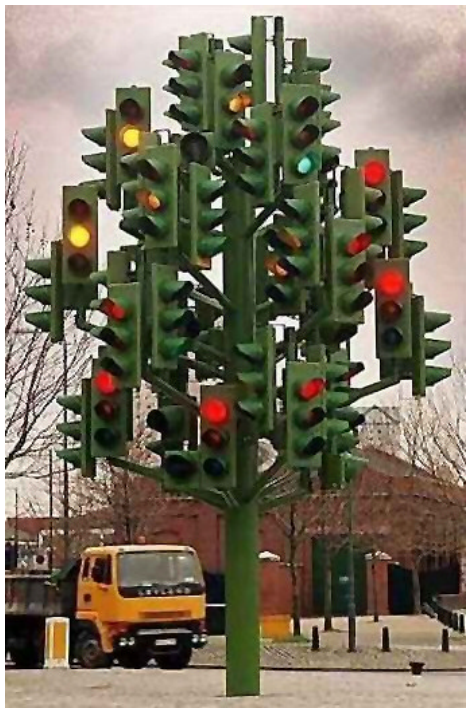
Um die Schaltung vor dem Bau zu simulieren und so ihre Funktionsweise zu überprüfen, wird ein Digitalsimulator verwendet. Wir verwenden die JAVA-Software LoGatSim[1]. Der Schaltungssimulator erlaubt eine repräsentative und übersichtliche Darstellung der Schaltungen durch individuelle Einfärbung der Leitungen und Leuchten. Die Software berücksichtigt ebenfalls die zeitliche Signalverzögerung der Bausteine, die in der Regel abhängig von der Zahl der intern verwendeten NAND-Gatter abhängig ist. Die physikalischen Nebenwirkungen einer rellen Schaltung lassen sich so besser voraussagen als bei einem rein logischen Simulator.

[1] LoGatSim Logik-Gatter-Simulator von Hans Peter Schneider, 2005, beigefügt in der Datei LoGatSim.jar.

Die Unterrichtsreihe

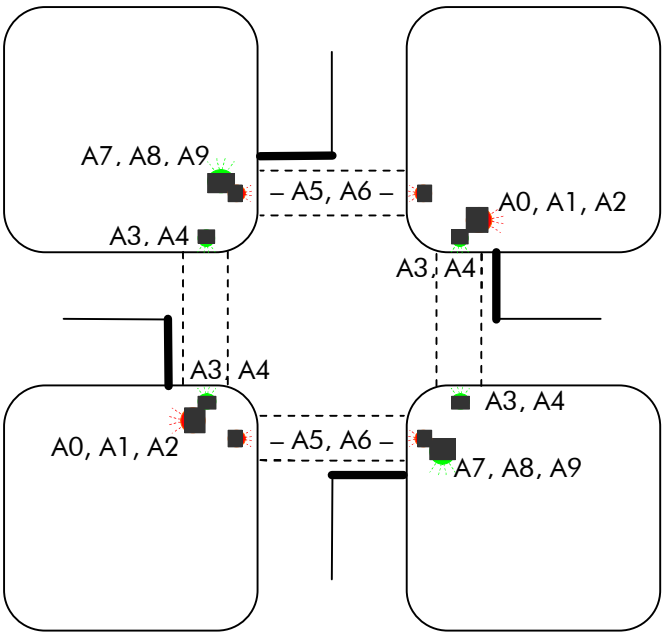
In dieser Unterrichtsreihe sollen die Schüler ihr Wissen zum ersten mal praktisch umsetzen und eine reelle Schaltung aus ICs löten. Sie verwenden die gelernten Theorien und Verfahren um die Schaltung zu entwerfen und einen Schaltplan zu erstellen, dann werden die nötigen Bausteine und Materialien gekauft und die Schaltung auf eine Experimentierplatine gelötet und schließlich die Schaltung in ein Modell eingebaut.

Aufgabenstellung



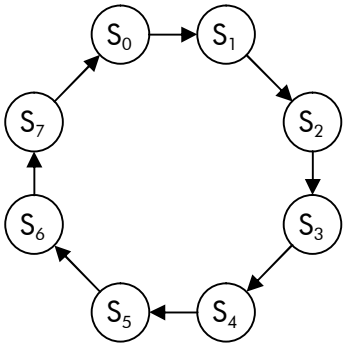
Die Schüler sollen jeder ein funktionsfähiges Modell einer Ampelanlage einer üblichen Straßenkreuzung bauen. Wie aufwändig dies betrieben wird, hängt von der Zeit und dem Engagement der Schüler ab. Erwartet wird, dass die Schüler zunächst die Funktionsweise einer Ampelanlage im reellen Betrieb beobachten, ausmessen und dokumentieren (Hausaufgabe über das Wochenende plus zwei Tage), um die Problemstellung und ihre Nebenbedingungen vollständig zu erfassen. Sie müssen sich dann auf eine Ampelanordnung festlegen, die sie modellieren wollen, denn es gibt im Straßenverkehr zahlreiche Varianten: separate Phasen und Ampeln für Abbieger oder Radfahrer, ereignisgesteuerte Ampeln mit Druckknopf oder Induktionsschleifen, unterschiedliche Zahl von einmündenden Straßen etc.

Wir betrachten hier das einfache Beispiel einer Kreuzung von zwei gleichwertigen Straßen mit einer Straßenampel und einer Fußgängerampel für jede der vier Fahrtrichtungen. Zwei jeweils entgegen gesetzte Fahrtrichtungen werden identisch geschaltet. Beobachtet man eine solche Ampel im Straßenverkehr, so stellt man fest, dass es bei jedem Wechsel eine kurze Phase gibt, in der alle Ampeln rot zeigen. Dies ist eine Sicherheitsvorschrift im Ampelbau und verhindert, das startende Autos in die letzten Autos des Querverkehrs fahren. Aufgabe der Messung einer Ampel ist die Feststellung solcher Eigenschaften und die Bestimmung der Zahl und Länge der verschiedenen Phasen eines vollständigen Zyklus'.



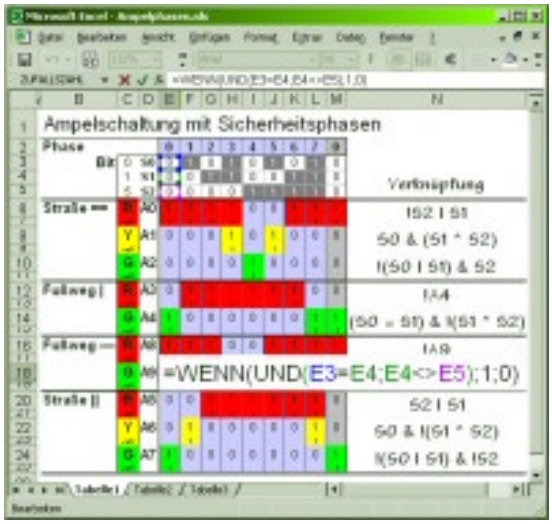
Jedes Licht ordne wir einem vom zehn Ausgangssignalen A0..A9 zu, das später von der Schaltung gesetzt werden und die Lichter ansteuern soll.

Der Automat



Der Zustandsgraph ist wenig interessant und leicht zu erstellen: Eine Hand voll Zustände in strenger Reihenfolge zyklisch angeordnet. Interessanter sind die zugehörigen Ausgaben, die die Ampellichter steuern. Da in einer Schaltung die Zustandsnummern nicht dezimal vorliegen, müssen die Ausgangswerte aus den binären Darstellungen der Zustandsnummern logisch berechnet werden.

In unserem Beispiel ergeben sich aus dem (sicheren) Wechsel der Grünphasen der beiden Straßenampeln acht (2^3, wie günstig!) Zustände.



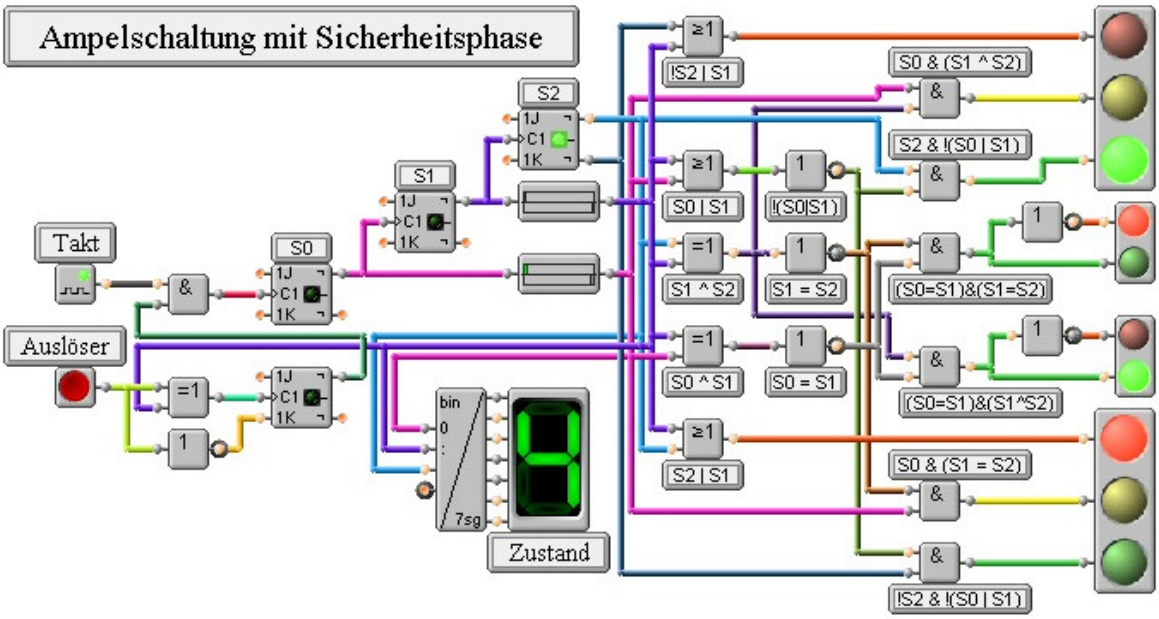
Die Phasen der Fußgängerampeln richten sich nach diesen. Wir stellen zunächst die Zustandstabelle mit den gewünschten Ausgabewerten auf und knobeln dann die für jeden Ausgang die zugehörige Berechnungsvorschrift aus. Anschließend tragen wir diese Verknüpfungsfunktionen in eine Excel-Tabelle ein und verifizieren unsere Funktionen wie oben beschrieben. Bei den Berechnungsvorschriften haben wir darauf geachtet, dass eine Verknüpfung zweier Bits möglichst oft mit der gleichen Operation vorkommt, da sie auch dann nur einmal berechnet werden muss. Dies gilt auffällig oft für die Verknüpfung (S1^S2), die zweimal direkt und zweimal negiert vorkommt.

Ampelschaltung mit Sicherheitsphasen

Phase			0	1	2	3	4	5	6	7	0	
Bit	0	S0	0	1	0	1	0	1	0	1	0	
	1	S1	0	0	1	1	0	0	1	1	0	Verknüpfung
	5	S2	0	0	0	0	1	1	1	1	0	
Straße ==	R	A0	1	1	1	1	0	0	1	1	1	!S2 | S1
	Y	A1	0	0	0	1	0	1	0	0	0	S0 & (S1 ^ S2)
	G	A2	0	0	0	0	1	0	0	0	0	!(S0 | S1) & S2
Fußweg |	R	A3	0	1	1	1	1	1	1	0	0	!A4
	G	A4	1	0	0	0	0	0	0	1	1	(S0 = S1) & !(S1 ^ S2)
Fußweg —	R	A8	1	1	1	0	0	1	1	1	1	!A9
	G	A9	0	0	0	1	1	0	0	0	0	(S0 = S1) & (S1 ^ S2)
Straße ||	R	A5	0	0	1	1	1	1	1	1	0	S2 | S1
	Y	A6	0	1	0	0	0	0	0	1	0	S0 & !(S1 ^ S2)
	G	A7	1	0	0	0	0	0	0	0	1	!(S0 | S1) & !S2

Die Simulation

Mit der Software LoGatSim wird der Automat dann als logische Schaltung erstellt. Für den Phasenwechsel benutzen wir einen Taktgeber. Die Grün- bzw. Rotphase soll jedoch länger dauern, daher wird eine Schaltung eingebaut, die den Taktgeber nach dem Wechsel blockiert, bis er von einem Taster wieder frei geschaltet wird. Der Phasenwechsel wird so durch den Taster ausgelöst. Die Freischaltung wird solange in einem Flipflop gespeichert, bis die Phase im Zustand 0 bzw. 4 erreicht ist Dieser Flipflop dient gleichzeitig als Entpreller für den Taster.



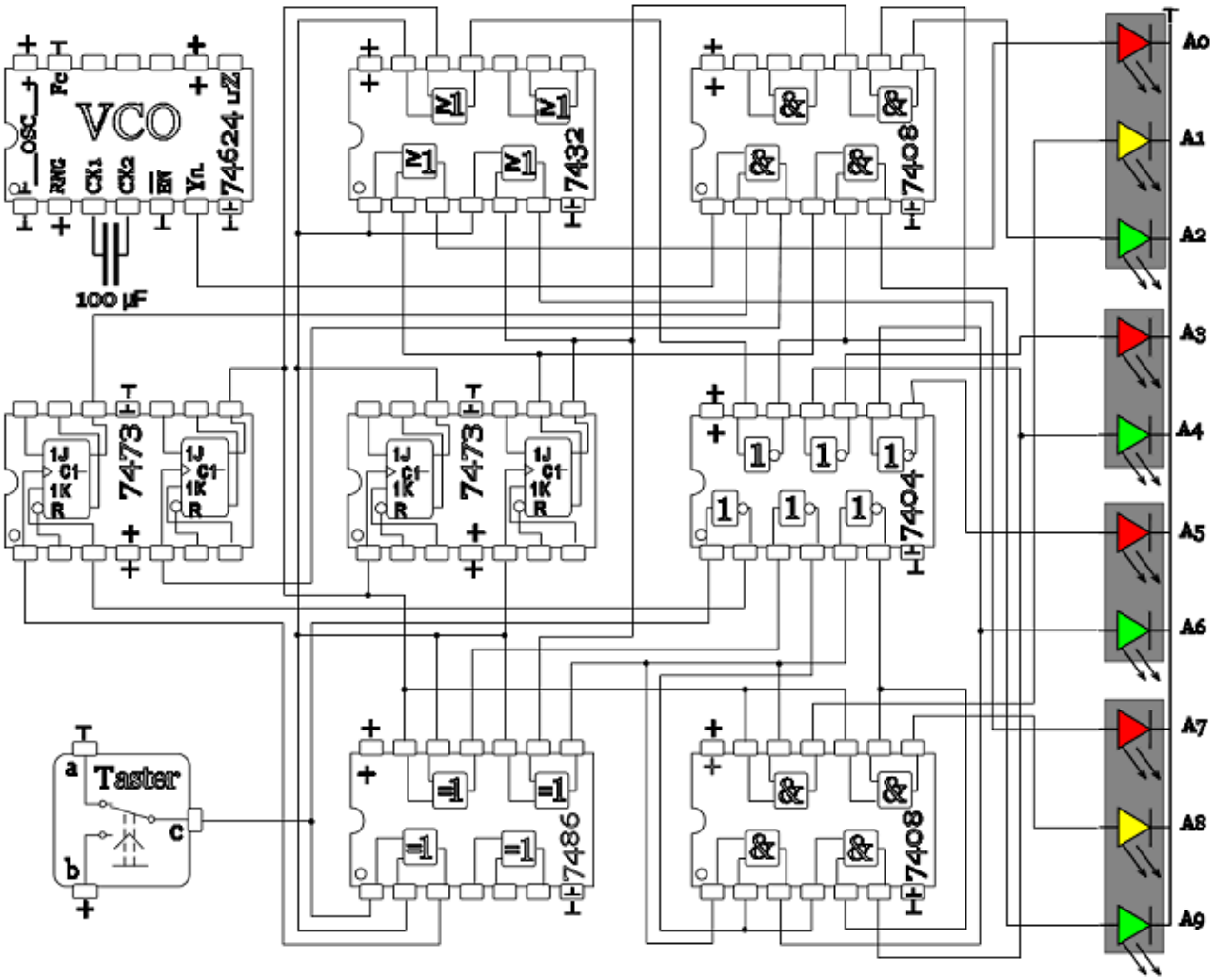
Drei Master-Slave-Flipflops erzeugen beim Phasenwechsel an ihrem Q-Ausgängen im Takt die Bitmuster der Zustandsnummern. Dann werden nach und nach die verschiedenen Signalkombinationen mit entsprechenden Gattern zusammengesetzt bis die Berechnungen der Ausgangssignale komplett sind. Mit Absicht werden die drei Operationen NOR NAND uns NXOR, da sie nur vereinzelt vorkommen, als zwei Gatter mit separaten NOT realisiert, um die ICs möglichst gut auszunutzen. Die zwei Delays verzögern die Signale der ersten beiden Flipflops S0 und S1, damit sie synchron zum S2-Signal wechseln und so das Flackern der Ampellichter beim Umschalten minimieren. In der reellen Schaltung können wir auf die Delays verzichten und das Flackern in kauf nehmen.

Die Schaltung

Die Simulation zeigt, dass die Schaltung stabil funktioniert, obwohl die Signalverzögerung der Flipflops beim Zustandswechsel zwischendurch kurz falsche Bitmuster erzeugen, die bei den Ampeln zwischendurch zum Flackern führen. Die Bestätigung durch die Simulation ermöglicht es uns, die verwendeten Gatter und Bausteine zu zählen und unsere Einkaufsliste für die nötigen TTL-Bausteine aufzustellen. Die Teile bereiten keine Schwierigkeiten und können schnell aus einer IC-Liste ausgewählt werden.

Bausteinliste	Einkaufliste
1 VCO	1 TTL-74624
4 JK-MS-Flipflops	2 TTL-7473
3 ODER-Gatter	1 TTL-7432
7 AND-Gatter	2 TTL-7408
3 XOR-Gatter	1 TTL-7486
6 NOT-Gatter	1 TTL-7404
(2 Delays)	(1 TTL-7431)
12 rote, 4 gelbe, 12 grüne LED	
1 Kondensator 100µF	
1 Platine, 8 IC-Sockel	
1 Spannungsquelle 5V	

Der nächste Schritt besteht darin, die Bauteile auf der Platine so anzuordnen, dass sie unter das Ampelmodell passt und Platz genug für die Verkabelung bleibt. Da wir Experimentierplatinen verwenden, müssen wir kein kompliziertes Layout entwerfen, wir verkabeln die ICs mit dünner Litze oder isoliertem Draht.



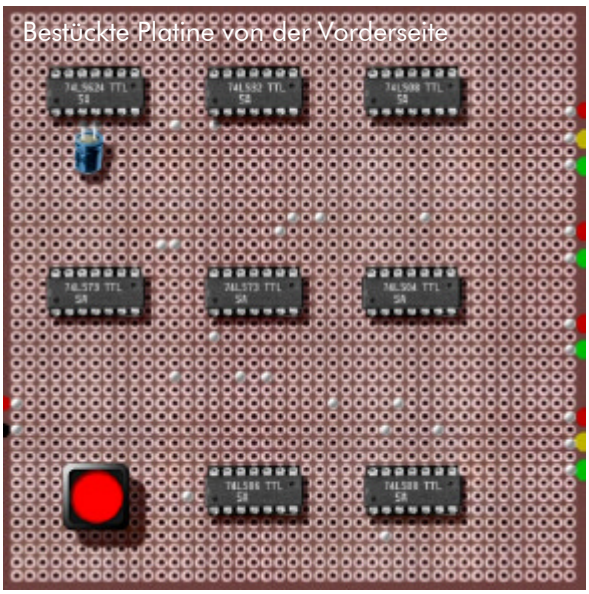
Um die Frequenz des VCO[2] festzulegen legen wir den FC-Eingang auf 0 und den RNG-Eingang auf 1, dadurch wird der Oszillator auf minimale Frequenz (ca. 1Hz = Sekundentakt) eingestellt. Möchte man die Frequenz regeln, schaltet man einfach einen Poti zwischen 1 und FC: Von 0 bis 5V Eingangsspannung regelt der FC-Eingang die Frequenz von ca. 1 bis 10Hz. Der Frequenzbereich des VCO hängt nachhaltig vom externen Kondensator ab, je kleiner die Kapazität, desto höher die Frequenz. Die Obergrenze liegt mit einem Kondensator von 15pF bei etwa 30MHz.

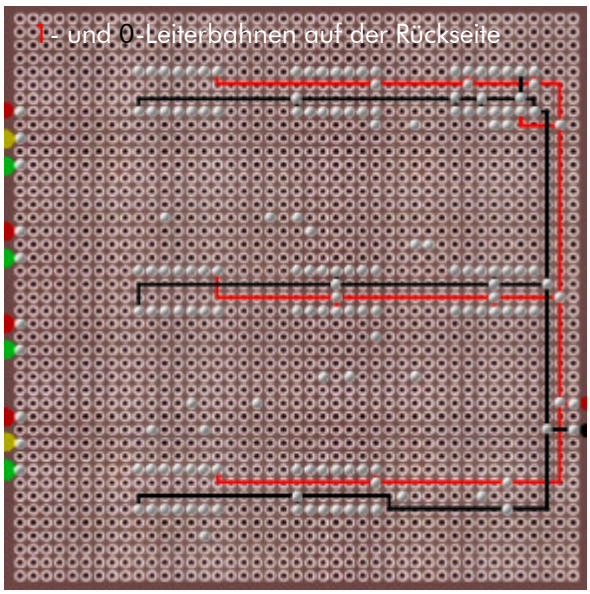
Der Taster muss so eingebaut werden, dass die geschaltete Leitung im Ruhezustand auf 0 liegt, weil sonst die anliegenden ICs die Leitung auf 1 halten. Will man die Kreuzung vollständig mit Ampeln ausstatten, liegen an jedem Ausgang zwei bzw. vier Dioden an. Daher sollte man die Ausgänge über Transistoren schalten, da die Dioden sonst zu viel Strom durch die ICs ziehen.

Damit ungeübte Schüler die ICs beim Löten nicht verbrennen, bekommt jeder IC einen Stecksockel verpasst. Zum Löten leiht man sich aus der Physiksammlung eine Hand voll Lötkolben und reserviert sich den Werk- oder Chemieraum (feuerfest sollte er sein). Praktischerweise beginnt man damit, die

[2] Nähere Angaben über die TTL74-VCO-Familie in der beigefügten Dokumentation 74LS624.pdf

Stecksockel und den Taster einzulöten, dann folgt der Kondensator. Dabei sollten die ICs etwa so weit von einander entfernt positioniert werden, dass so viele Kontaktreihen zwischen ihnen frei bleiben, wie Leiter hindurch führen. Die Stromversorgung kommt dann als nächstes dran und wird auf der Rückseite unter den ICs entlang geführt, da sind sie den anderen Leitungen später am wenigsten im Weg. Erst dann werden, auch auf der Rückseite, die Leiterbahnen der Schaltung verlegt, am besten mit verschiedenfarbig isolierten, festen Kupferdrähten, die ihre Lage und form behalten, damit man den Überblick nicht verliert. Zu letzt bringt man auf der Vorderseite an den bunt markierten Stellen am Platinenrand die Litzen an, die zur Stromquelle und zu den Ampeln führen. Bis die Schüler Kontakte auf der Platine so sauber löten können, dass keine ungewollten Verbindungen benachbarter Kontakte entstehen, dauert es etwas. Am besten ist, wenn die Schüler das Löten auf einem Platinenrest, der beim Zuschneiden übrig geblieben ist, mit den Leiterdrähten üben, bis sie selbst mit dem Ergebnis zufrieden sind.





Das Modell

Die Straßenkreuzung selbst wird dann auf eine Platte gebaut, die die gleiche Größe wie die quadratische Platine hat und mit vier langen Schrauben und Muttern in den vier Ecken so weit über der Platine angebracht wird, dass der Taster in die obere Platte integriert werden kann. Diese Schrauben halten auch die Platine nach unten in ausreichendem Abstand über der Tischplatte. Für Demonstrationszwecke ist es aber auch effektvoll, die Schaltungsplatine nicht zu verstecken, sondern senkrecht hinter dem Modell so anzubringen, dass beim Drehen des Modells Vorder- und Rückseite der Platine gut zu betrachten sind.

Wem es zu mühsam ist, für die Dioden die Ampeln selbst zu basteln, der kann sich im Modellbau auch Ampeln mit verkabelten Dioden fertig oder im Bausatz zum zusammenstecken kaufen (Modelleisenbahnmaßstab H0 oder N), allerdings sind diese Ampelbausätze mit Preisen zwischen 9,– und 20,– € teurer als der ganze Rest der Schaltung. Zu empfehlen ist aber Rasen, Straßenasphalt, Straßenmarkierungen, Autos, Menschen etc. im Modellfachhandel zu kaufen, wenn man die Kreuzung mit Details ausschmücken möchte; meist reicht da jeweils eine Packung für die ganze Klasse.

Viel Spaß beim Nachmachen!

Quellen

[1] „Technische Informatik – Hardware“, Georg Beckmann, VLIN 2002

[2] „Voltage-Controlled Oscillators – SN54LS624 thru SN54LS629, SN74LS624 thru SN74LS629“ Spezifikation und Datenblatt, Texas Instruments, Dallas 1980, revised 1988