

Grundlagen der Rechnerarchitektur - Labor

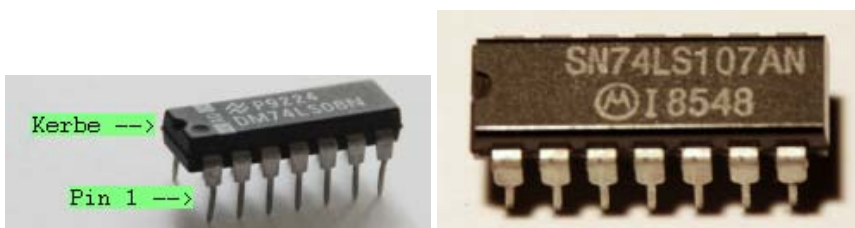
Im Laufe dieses Semesters bauen die Versuche alle aufeinander auf. Man sollte deshalb zu jedem Versuch nicht nur die aktuelle Anleitung zur Hand haben, sondern auch die der Versuche davor.

Versuch 1: Kombinatorische Logik

In diesem Versuch wird kombinatorische Logik betrachtet. Integrierte Schaltungen (Integrated Circuits, ICs) mit einzelnen Logikgattern werden entsprechend der booleschen Algebra bis zu einer Arithmetik-Logik-Einheit (Arithmetic-Logic-Unit, ALU) zusammengeschaltet.

In dieser Anleitung wird zuerst das Experimentiermaterial vorgestellt, dann der allgemeine Ablauf eines Versuchs beschrieben und einige allgemeine Hinweise gegeben. Danach folgen die Aufgaben. Die letzte Seite ist ein "Spickzettel" für die ICs, die im Praktikum benutzt werden können.

Experimentiermaterial: ICs



Die Bilder zeigen ICs aus dem Praktikum. Aus einem Kunststoffgehäuse führen 14 Beine heraus, die Pins genannt werden. Am Kunststoffgehäuse ist eine Kerbe angebracht, die die Orientierung des ICs festlegt. (Der Aufdruck ist nicht maßgeblich für die Orientierung, nur die Kerbe!) Wenn man nun von oben auf das IC schaut, also die Pins nach unten zeigen und die Kerbe links ist, dann ist links unten Pin Nummer 1. Gegen den Uhrzeigersinn werden die Pins dann weitergezählt, so daß oberhalb der Kerbe, also links oben Pin 14 liegt. Die Regel gilt so auch bei ICs mit anderer Pinanzahl, allerdings ist der letzte Pin dann natürlich nicht mehr Nummer 14 sondern z.B. 16.

Außer den Pinnummern muß man natürlich noch wissen, was in einem IC für Funktionen realisiert sind. Hierzu muß man den Aufdruck lesen. Das linke Foto zeigt einen 74LS08, das rechte einen 74LS107, der Rest des Aufdrucks ist im Praktikum nicht von Interesse. Ein wichtiges Detail ist dabei, daß die Zahl hinter dem LS komplett gelesen werden muß. Ansonsten wäre ein 74LS10 nicht von einem 74LS107 zu unterscheiden. Auf der letzten Seite dieser Anleitung ist ein Überblick über die im Praktikum vorkommenden ICs gegeben. Dort findet man z.B. den oben gezeigten 74LS08 als IC, das vier Und-Gatter mit je zwei Eingängen enthält. Das Schema enthält die Pin-Nummern und die Kerbe, damit die Zuordnung einwandfrei möglich ist.

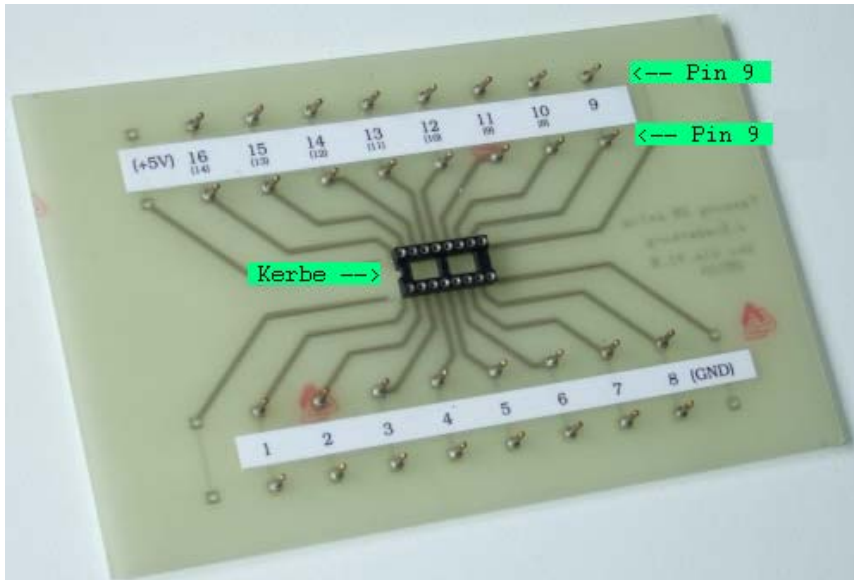
Alle im Praktikum verwendeten ICs benötigen eine stabilisierte Betriebsspannung von 5 Volt. Diese wird über die Pins "Vcc" und "GND" zugeführt, und zwar die +5 Volt an Vcc und 0 Volt an GND.

Bei den im Praktikum verwendeten ICs gilt außerdem die Konvention, daß eine logische 0 durch ein Spannung von 0 bis 0,8 Volt, eine logische 1 durch 2,4 bis 5 Volt dargestellt wird. Die ICs reagieren also auf Spannungen an den Eingängen entsprechend dieser Konvention und geben ebenso geeignete Spannungswerte aus. Auf diese Weise "verstehen" sich die ICs und als Entwickler kann man mit logisch 0 oder 1 arbeiten, ohne ständig mit Spannungs- und/oder Stromwerten rechnen zu müssen. Außerdem ermöglicht diese Konvention, daß die Betriebsspannung (+5V, VCC) als Konstante 1, Masse (0V, GND) als Konstante 0 benutzt werden kann.

Ausgänge dürfen nie zusammengeschaltet werden, da die Spannungswerte für log. 0 und 1 jeweils aktiv erzeugt werden. Deshalb kann man nicht erwarten, daß 5 Volt und 0 Volt addiert zu 5 Volt werden. Vielmehr entsteht beim Zusammentreffen einer 1 und einer 0 an zusammengeschalteten Ausgängen ein Kurzschluß. Die beteiligten ICs werden dann sehr heiß und werden nach kurzer Zeit kaputt gehen. Wenn sich ein Signal aus mehreren Eingangssignalen zusammensetzen soll, dann müssen diese über Gatter der gewünschten Funktion zusammengeführt werden.

Offene Ausgänge sind grundsätzlich unproblematisch. Offenen Eingänge gelten bei den im Praktikum verwendeten ICs als mit log. 1 beschaltet, allerdings sind moderne ICs so gut, dass statische Ladungen der Umgebung bereits dazu führen können, dass ein offener Eingang als log. 0 erkannt wird, wenn auch meist nur sehr kurz. Bei rein kombinatorischen Schaltungen ist dies nicht schlimm. Flip-Flops hingegen sehen auch kürzeste Impulse als Taktsignal und schalten daher "zufällig". Es empfiehlt sich daher bei Schaltungen mit Flip-Flops alle Eingänge mit log. 0 (0 Volt) oder 1 (+5 Volt) geeignet zu beschalten. Spätestens wenn die Schaltung "spinnt" sollte man kontrollieren, ob offene Eingänge irgendwo in der Schaltung vorhanden sind. Dies gilt insbesondere auch für Eingänge an Flip-Flops, die man eigentlich gar nicht braucht.

Experimentiermaterial: Leerplatinen



Diese dienen der Aufnahme der ICs, um diese elektrisch miteinander verbinden zu können. Auf jeder Leerplatine befindet sich ein IC-Sockel und pro Pin des Sockels zwei elektrisch damit verbundene Lötnägel. Zum Experimentieren wird ein IC in die Fassung gesteckt und dann Verbindungen zu anderen ICs über Kabel hergestellt, die auf die Lötnägel gesteckt werden.

Die Fassungen haben Kerben, die denen der ICs entsprechen. D.h., die ICs müssen so eingesteckt werden, daß die Ausrichtung der Kerben übereinstimmt. Beim Einsetzen von ICs ist darauf zu achten, daß die Pins nicht verbogen werden. Zum Herausnehmen der ICs dient ein dem Versuchsmaterial beiliegender Schraubendreher. Diesen schiebt man zwischen IC und Fassung, so daß das IC herausgehoben wird. Auf diese Weise wird erreicht, daß die IC-Pins nicht verbogen werden.

Jeder IC-Pin ist mit zwei Lötnägeln verbunden. Im Bild oben sind die beiden mit Pin 9 der Fassung verbundenen Lötnägel gekennzeichnet. Die zwei Lötnägel ermöglichen den Anschluß einer Signalleitung an einen IC-Pin und das gleichzeitige Weiterführen des Signals.

Es stehen Leerplatinen für ICs mit 14 und 16 Anschlußpins zur Verfügung. Sollten für die Bearbeitung einer Aufgabe die Leerplatinen mit der benötigten IC-Größe nicht ausreichen, können ICs auch in Leerplatinen mit mehr Anschlußpins gesetzt werden, also ein 14-poliges IC in eine 16-polige Fassung. Für diesen Zweck stehen auf den 16-poligen Leerplatinen auch die Pinnummern in Klammern, die gelten, wenn man Pin 1 eines 14-poligen ICs in Pin 1 der Fassung setzt. Es ist dann besonders darauf zu achten, daß die Spannung korrekt angeschlossen werden muß, weil Pin 8 der Leerplatine nicht mehr der GND-Pin des ICs ist. Sollten trotz dieser Maßnahme zu wenige Platinen vorhanden sein, dann muß der Schaltungsentwurf geändert werden. Es ist grundsätzlich nicht zulässig weitere Platinen zu benutzen!

Experimentiermaterial: Chipsimulatorplatinen

Statt der Leerplatinen gibt es auch Platinen, die alle ICs des Praktikums simulieren können. Der Einsatz dieser Platinen ist erst ab Aufgabe 4 dieses ersten Versuchs zulässig!



Zur Simulation eines ICs stellt man (mit dem Schraubenzieher vorsichtig) an den Drehschaltern S1 und S2 das gewünschte IC ein. Die Schalter werden genau beim Einschalten des Stroms ausgewertet, und danach verhält sich die Platine entsprechend, bis der Strom wieder ausgeschaltet wird. Die Drehschalter wählen die ICs entsprechend der Tabelle, die unten am Platinenrand aufgedruckt ist:

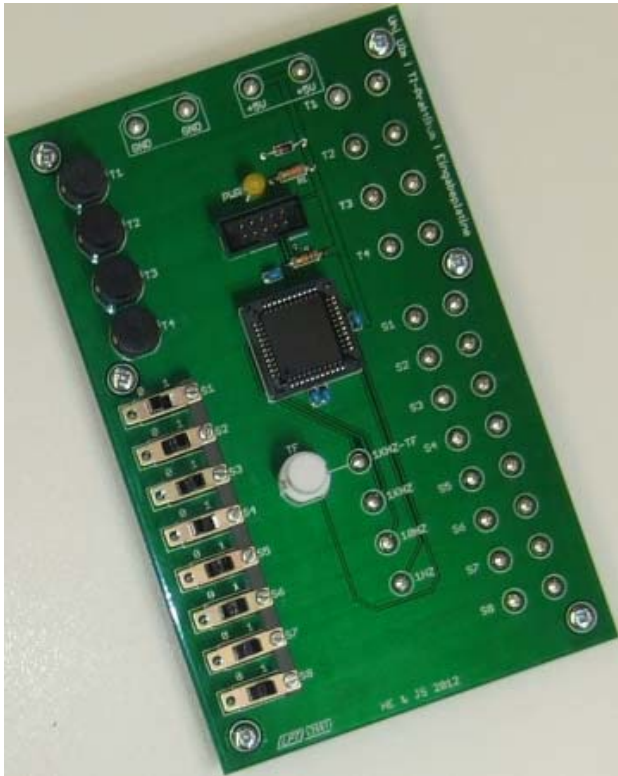
| | | | | | |
|--------|--------|-----|-----|-----|-----|
| 0 AND | 4 CNT | 160 | 161 | 168 | 169 |
| 1 OR | 5 MISC | - | 84 | 86 | 266 |
| 2 NAND | 6 FF | 74 | 187 | 95 | 164 |
| 3 NOR | 7 MUX | 138 | 151 | - | - |

- Die linke Hälfte dieser Tabelle zeigt die Einstellungen 0 bis 3 für S1. Hier werden nur ICs mit Gattern UND, ODER, NAND und NOR simuliert. S2 wählt hier immer die Anzahl der gewünschten Eingänge pro Gatter, im Bereich 2 bis 5. Beispiel: Will man NAND-Gatter mit 3 Eingängen, dann stellt man S1=2 und S2=3 ein. (Dies ergibt das IC 74LS10.) Man kann hier sogar ICs einstellen, die es real gar nicht gibt, wie z.B. zwei Oder-Gatter mit je 5 Eingängen.
- Die rechte Hälfte der Tabelle gibt an, welches IC simuliert wird, wenn man S1 auf 4, 5, 6 oder 7 stellt (wählt die Tabellenzeile), und was dann S2 bei Stellung 0 bis 3 auswählt. Beispiel: Für S1=5 und S2=2 steht in der Tabelle "86". Es wird also das IC 74LS86 mit 4 Exor-Gattern simuliert.

Grundsätzlich simuliert die Platine die ICs immer Pin-genau, wobei folgende Zusatzregeln gelten:

- Die Stromversorgung der Chipsimulatorplatine erfolgt immer über Pin 8 und Pin 16, insbesondere auch dann, wenn 14-polige ICs simuliert werden. Noch besser ist es, die Pins an der Platinenoberkante zu benutzen. In letzterem Fall kann man die Kontakte der Pins 8 und 16 gut für Logik-Konstanten 0 und 1 nutzen.
- Für ICs mit 16 Pins stimmen die Pinnummern ohne Klammern, während bei 14 poligen ICs die Zahlen in Klammern am Platinenrand gelten, sofern angegeben. (D.h., Pin 9 und 7, ohne Klammern, sind funktionslos, wenn 14-polige ICs simuliert werden.)
- Bei der Simulation des ICs 74LS02 sind die Gatter nicht korrekt dem Chip entsprechend angeordnet, sondern so wie z.B. beim IC 74LS00 oder jedem anderen IC, das vier Gatter mit je zwei Eingängen hat.
- Wählt man Gatter mit fünf Eingängen, dann ist das Pinout entsprechend so, als ob man ein IC mit nur vier Eingängen hat, aber der jeweils freie Pin ein weiterer Eingang des Gatters ist. D.h., Pin 3 wirkt auch auf den Ausgang Pin 6, und 11 entsprechend auf Pin 8.

Experimentiermaterial: Eingabeplatine



Diese Platine wird zum Generieren von Signalen verwendet. Es stehen vier Taster (T1 bis T4) und acht Schalter (S1 bis S8) zur Verfügung, die ihr Signal jeweils auf zwei Lötnägel ausgeben. Zusätzlich gibt es vier Frequenzgeneratoren, die ihr Signal auf jeweils einen Lötnägel in der Platinenmitte ausgeben.

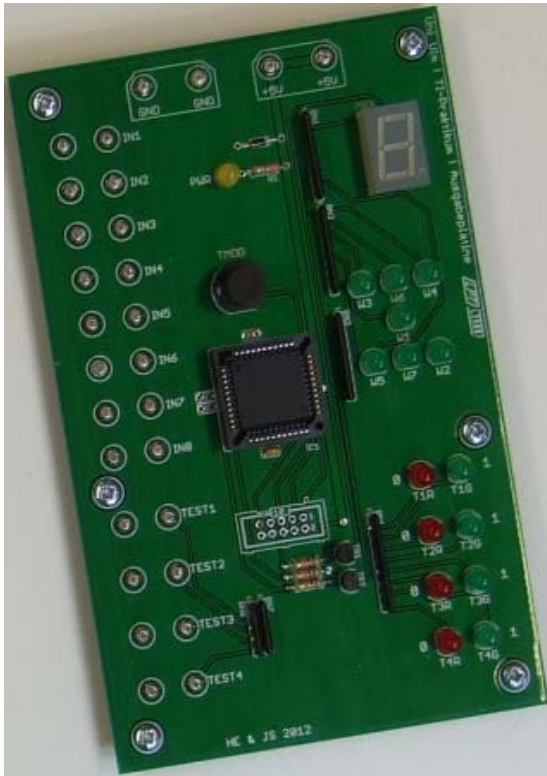
Die Platine benötigt genau wie einzelne ICs 5 Volt Betriebsspannung. Diese muß, entsprechend der Beschriftung, am oberen Platinenrand angeschlossen werden.

Die Ausgänge der vier Taster sind mit T1 bis T4 bezeichnet. An diesen liegt bei gedrücktem Taster logisch 1, sonst logisch 0 an. Die Taster sind entprellt, d.h., sie geben pro Tastendruck nur einen Impuls aus. Ohne Entprellung gibt ein mechanischer Taster beim Wechseln zwischen gedrückt und losgelassen evtl. mehrere Wechsel zwischen 0 und 1 aus, bis der neue Zustand stabil ausgegeben wird.

An den Schalterausgängen S1 bis S8 liegt je nach Schalterstellung entweder 0 (Schalthebel nach rechts) oder 1 (Schalthebel nach links) an. Die Schalter sind ebenfalls entprellt.

Die Frequenzausgänge liefern, entsprechend der Beschriftung auf der Platine, 1 Hz, 10 Hz, 1 kHz und noch einmal 1 kHz. Letzterer Ausgang liefert im Ruhezustand nur eine log. 0, und die Frequenz von 1 kHz erst, wenn man die zugehörige Taste drückt. Falls Ihnen die Einheit Hz bzw. kHz nichts sagt: 1 Hz bedeutet ein mal pro Sekunde hin- und zurückschalten (hier von 0 auf 1 und zurück). 10 Hz entsprechend 10 mal, usw.

Experimentiermaterial: Ausgabeplatine



Mit dieser Platine werden Signale angezeigt. Im unteren Drittel sind vier Logiktester vorhanden, darüber ist eine Würfelanzeige und darüber eine 7-Segment-Anzeige.

Die Spannungsversorgung der Platine muß wie immer mit einem roten Kabel an einen der mit "+5V" bezeichneten Lötstifte angeschlossen werden, und an einem der "GND"-Lötstifte muß 0V mit einem schwarzen Kabel angeschlossen werden.

Die Ausgabeplatine besitzt nur Eingänge (IN1 bis IN8 und Test1 bis Test4), jeweils mit zwei Lötnägeln. Wie immer dient der zweite Lötnagel dem Zweck, dass das Eingangssignal, bei Bedarf noch an weitere Eingänge geführt werden kann.

Die Logiktester (Test1 bis Test4) sind unabhängig voneinander und auch unabhängig von 7-Segment-Anzeige und Würfel. Sie können also jederzeit benutzt werden als allgemeine Anzeigen für Logikpegel, insbesondere auch zur Fehlersuche. Ausnahmsweise werden drei Eingangszustände unterschieden: Wenn die rote LED leuchtet, bedeutet dies, daß das Signal 0 anliegt. Leuchtet die grüne LED, wird signalisiert, daß 1 anliegt. Sind beide LEDs dunkel, zeigt dies an, daß der Eingang offen ist. Dies kann auch auf eine Verkabelung hindeuten, die nur Eingänge zusammenschaltet. Sollten scheinbar beide LEDs leuchten, so deutet dies auf sehr schnelle Wechsel zwischen 0 und 1 am Eingang hin.

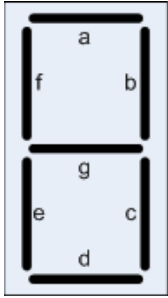
Die Auswahl des Anzeigemodus zwischen den 7-Segment-Anzeige-Modi und der Würfelanzeige erfolgt mittels der Taste. Drückt man einmal auf die Taste, so wird für kurze Zeit der aktuelle Modus angezeigt. Drückt man während dieser Zeit die Taste (auch mehrfach) erneut, so wechselt man zyklisch durch die Modi:

- 7-Segment-Anzeige komplett von IN1 bis IN4 ansteuern
- Segment 'a' der 7-Segment-Anzeige von IN8 ansteuern
- Segment 'b' der 7-Segment-Anzeige von IN8 ansteuern
- ...
- Segment 'g' der 7-Segment-Anzeige von IN8 ansteuern
- Den Dezimalpunkt der 7-Segment-Anzeige von IN8 ansteuern
- Den Würfel ansteuern

Im ersten Modus ist die 7-Segment-Anzeige ausgewählt. Die Eingänge IN1, IN2, IN3 und IN4 werden als Binärzahl mit den binären Wertigkeiten $2^0 \dots 2^3$ interpretiert, und der entsprechende Zahlenwert als menschlich lesbare Hexadezimalziffer 0, 1, 2, ..., 9, A, B, ..., F dargestellt.

Die folgenden acht Modi tun dies ebenfalls, jedoch wird eines der Segmente der 7-Segment-Anzeige weggelassen. Dieses eine Segment (oder der Dezimalpunkt) wird stattdessen direkt über den Eingang IN8 angesteuert.

Das folgende Diagramm zeigt die Benennung der Segmente der 7-Segment-Anzeige von a bis g:



Falls die Würfel-Anzeige ausgewählt ist, so werden deren LEDs über die Eingänge IN1 bis IN7 betrieben. Die Zuordnung der Eingänge zu den LEDs ist wie folgt:

| | | |
|-----|-----|-----|
| IN3 | IN6 | IN4 |
| | IN1 | |
| IN5 | IN7 | IN2 |

Experimentiermaterial: Netzteil & Verbindungskabel

Alle ICs, die Ein- und die Ausgabeplatine müssen mit Strom versorgt werden. Die dafür notwendigen Verbindungen sind die einzigen, die nicht direkt der entworfenen Logik dienen. Da diese Leitungen deshalb "etwas besonderes" sind gibt es auch Konventionen in der ganzen Elektrotechnik dafür:

- **Masseleitungen sind immer schwarz.** (Masse == GND == 0 Volt == Minus)
- Die (wichtigste) **Versorgungsspannung ist immer rot.** (+5 Volt == Vcc == Plus)

Die Farben beziehen sich auf die Isolation der Leitungen. Im Praktikum wird verlangt, daß diese Konvention eingehalten wird, daß also Masseleitungen immer in schwarz ausgeführt werden müssen, und auch, daß andere Leitungen nicht schwarz sein dürfen! Ebenso, daß nur Rot die Versorgungsspannung von +5 Volt führt. Eine der Leitungen des Netzteils ist mit einem roten Stück Isolation versehen, womit klargestellt ist, daß diese Leitung die +5 Volt Leitung ist.

Alle Platinen werden parallel an das Netzteil angeschlossen. Da am Netzteil nur jeweils ein rotes und ein schwarzes Kabel vorhanden ist, muß der Anschluß für die weiteren Platinen von der zuerst angeschlossenen Platine erfolgen. (Natürlich mit roten und schwarzen Kabeln.) Für alle anderen Verbindungen, also Logik-Leitungen, die nicht direkt zum Netzteil führen, können die Kabel in den anderen Farben beliebig genutzt werden.

Sollte im Praktikumsbetrieb einmal eine rote und eine schwarze Leitung zusammenkommen, so passiert nichts schlimmes. Die Netzteile sind kurzschlußfest, d.h., sie schalten ab, wenn es zum Kurzschluß kommt, und sie schalten auch sofort wieder ein, wenn der Kurzschluß beseitigt ist.

Soll ein Eingang mit einer Konstanten 0 oder 1 beschaltet werden, so kann, wie oben bereits gesagt, die Versorgungsspannung dafür genutzt werden. Hier tritt die Frage auf, ob eine solche Leitung, die z.B. +5 Volt für eine log. 1 auf einen Eingangs-pin schaltet eine Versorgungsspannungsleitung oder eine Signalleitung ist, d.h., muss hier rot oder eine andere Farbe benutzt werden? Kurz gesagt: das ist egal. Wichtig ist, dass eine rote Leitung immer +5 Volt führt und eine Schwarze immer 0 Volt, und dass alle Leitungen mit Versorgungsstrom dem Farbcode gehorchen. "Versorgungsstrom" sind die Leitungen, die dafür sorgen, dass alles läuft, und nicht der booleschen Logik dienen.

Die Kabel sind gelötet. Um die Lebensdauer zu erhöhen, sollten die Kabel beim Stecken und Abziehen nur am Stecker gehalten werden. Sollte ein Kabel sich beim Stecken sehr locker anfühlen, dann sollte man das Kabel aussortieren zur Reparatur. Eine lockere Verbindung könnte sich von alleine Lösen, was einen schwer zu findenden Fehler darstellt.

Ablauf eines Versuchs

Zuerst muß ein vollständiges Schaltbild gezeichnet werden. Ohne dieses wird kein durchgeführter Versuch akzeptiert! Aus diesem Schaltbild bestimmt sich dann die Art und Anzahl der benötigten ICs. Diese werden auf Leerplatinen gesetzt bzw. bei Chipsimulator-Platinen die Schalter passend eingestellt. Da die ICs jetzt feststehen, können nun im Schaltbild alle IC-Anschlüsse mit Pin-Nummern versehen werden. Im Schaltbild dürfen jetzt lediglich die Versorgungsspannung und unbenutzte Gatter noch fehlen. Alle anderen Anschlüsse der benutzten ICs müssen eingezeichnet sein, d.h., wenn beispielsweise ein NAND-Gatter als Inverter benutzt werden soll, indem alle Eingänge des NAND-Gatters zusammengeschaltet werden, so müssen trotzdem alle NAND-Gatter-Anschlüsse im Schaltbild einzeln erkennbar sein.

Danach wird die Versorgungsspannung zu allen Platinen geführt (mit roten und schwarzen Kabeln), das Netzteil darf aber noch nicht eingesteckt werden! Anschließend wird eine Datenleitung nach der anderen gesteckt, die dann auf dem Schaltbild abgehakt wird. Auf diese Weise sollten falsche Verbindungen praktisch nicht entstehen.

Zuletzt wird die Versorgungsspannung eingeschaltet, d.h., das Netzteil in die Steckdose gesteckt. Verhält sich die Schaltung nicht wie geplant, können folgende Maßnahmen weiterhelfen:

- **Versorgungsspannung ausschalten**

Das nur kurze Betreiben einer Schaltung verhindert (meist), daß ICs zerstört werden, die falsch beschaltet sind. Für weitere Untersuchungen wird die Versorgungsspannung dann immer nur kurz eingeschaltet.

- **Heiße ICs**

Falsch beschaltete ICs kann man oft auch daran erkennen, daß sie sehr heiß werden (sofort ausschalten!). Heiße ICs deuten darauf hin, daß ein Ausgang direkt an die Versorgungsspannung (+5V oder GND) angeschlossen ist, zwei direkt miteinander verbundene Ausgänge vorliegen oder ein IC falsch herum in der Fassung sitzt. Im Praktikum kommen keine ICs vor, die mehr als handwarm werden dürfen.

- **Sichtkontrolle**

Sind alle IC-Fassungen mit einem IC bestückt? Sind die richtigen IC-Typen eingebaut (gerne werden 74LS10 und 74LS107 verwechselt)? Sitzen die ICs richtigerum in den Fassungen? Bei ICs, die in größeren Fassungen sitzen: sind die richtigen Pins beschaltet, speziell die Versorgungsspannung? Sind alle Schaltungsteile (ICs, Eingabe- und Ausgabeplatine) mit Strom versorgt (rote und schwarze Kabel verfolgen)?

- **Überprüfen von Teilschaltungen**

Zwischenergebnisse können mit Hilfe der Logiktester auf der Ausgabeplatine sichtbar gemacht werden. Aus dem Schaltbild kann man leicht ersehen, an welchem Pin von welchem IC welches Zwischenergebnis vorkommen muß. Man überlegt also, wie dieses Zwischenergebnis aussehen muß und vergleicht dann mit dem Verhalten der Schaltung.

- **Aus dem eigenen Schaltbild die realisierte Funktion ableiten**

Ergibt sich aus dem entworfenen Schaltbild nicht die Funktion, die realisiert werden sollte ("Rückwärtsanalyse"), so liegt ein logischer Entwurfsfehler vor. Oft ist es dann angebracht, den Entwurf ganz von vorne zu beginnen und nach jedem Schritt zu vergleichen, ob beim ersten Entwurf dasselbe Zwischenergebnis vorlag.

- **Den Betreuer fragen**

Dies sollte erst dann geschehen, wenn alle bisher genannten Maßnahmen wirkungslos geblieben sind.

Bei jeder Änderung der Schaltung ist vorher die Versorgungsspannung auszuschalten. Vor dem Wiedereinschalten sollten alle Änderungen nochmals überprüft werden. Eine Änderung besteht meist aus dem Hinzufügen oder auch dem Entfernen von Leitungen. Speziell letzteres wird oft vergessen.

Wenn im Versuch kein spezieller Hinweis enthalten ist, so kann (und sollte) danach der Versuch vollständig abgebaut werden. Hierfür ist zuerst die Versorgungsspannung auszuschalten, danach werden alle Kabel entfernt. Speziell die ICs sollten aus den Fassungen entfernt werden, da ein fehlendes IC im nächsten Versuch nicht kaputtgehen kann, ein falsches dagegen schon! Lediglich die Versorgungsspannung kann aus Bequemlichkeit meist stehen bleiben. Dann ist aber Vorsicht geboten, speziell wenn später eine andere IC-Größe (Pin-Anzahl) in einer Fassung benutzt wird. Deshalb sollte man den Schritt "Versorgungsspannung anschließen" immer bewußt vollziehen.

Sonstiges, Anmerkungen, Tipps

In diesem Praktikum wird weitgehend von der elektronischen Basis abstrahiert. Dies bedeutet, daß z.B. das Zeitverhalten und auch die Strombelastbarkeit der IC-Ausgänge im Praktikum fast oder gar nicht behandelt werden.

Bei allen Versuchen gilt, daß kein Material benutzt werden soll, das nicht vorgegeben ist. Das bedeutet beispielsweise, daß ein Schaltungsentwurf, der 7 ICs und damit 7 Leerplatinen/Chipsimulatorplatinen benötigen würde, im Praktikum nicht als Lösung akzeptiert wird. Diese Beschränkung wird nur bei den wenigsten Versuchen eine echte Einschränkung darstellen, ist aber praxisnah, da zusätzliche Teile jeden Entwurf unnötig verteuern und auch fehleranfälliger machen.

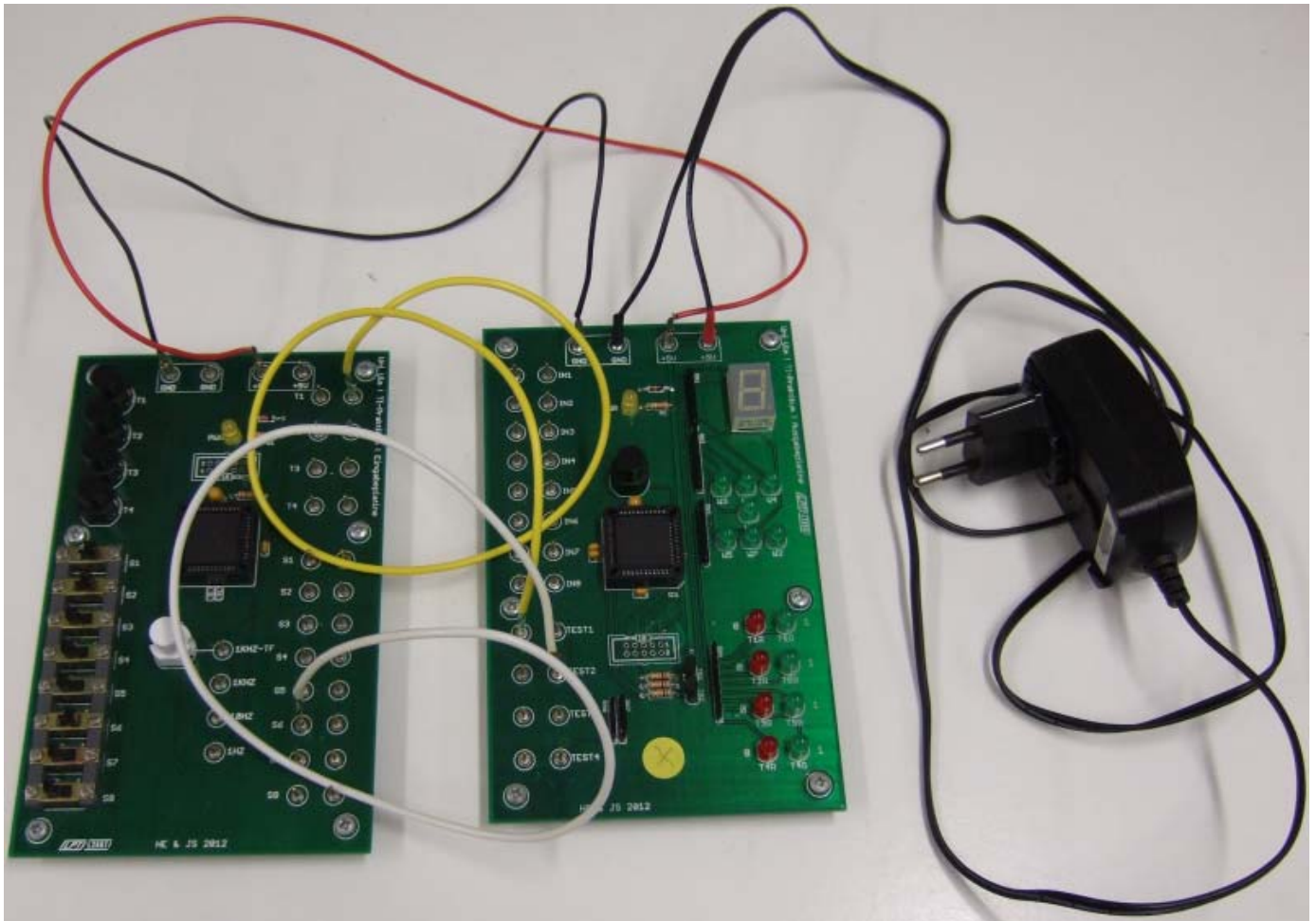
Die Beschränkung auf nur wenige ICs wird allerdings zum Problem, wenn man jede Operation einer booleschen Funktion "blind" in Hardware umsetzt. Statt dessen empfiehlt es sich, unbenutzte Gatter der bereits gewählten ICs für andere Operationen zu benutzen. Beispiele hierfür sind:

- freie NAND-, NOR- und EX-OR-Gatter können zu Invertern werden.
- NAND und UND entsprechen einander bis auf den invertierten Ausgang. Ebenso NOR und ODER. Mit einem unbenutzten Gatter, das Inverter werden kann, gewinnt man hier oft.
- "Zu viele" Eingänge an einem Gatter können immer "totgeschaltet" werden. D.h., ein NAND mit drei Eingängen kann als eines mit zwei Eingängen dienen. (Überflüssige OR/NOR-Eingänge auf 0, UND/NAND auf 1.)
- Ein UND kann auch als ODER benutzt werden (DeMorgan!) und auch umgekehrt ein ODER als UND. Die dabei notwendig werdenden Inverter sind relativ oft schon vorhanden oder entfallen beim weiteren Umformen wieder.

Defektes Material ist den Betreuern zu melden. Diese werden umgehend für Ersatz sorgen, um auch nachfolgenden Gruppen problemloses Arbeiten zu ermöglichen. Alle defekten Materialien werden, soweit möglich, repariert. Bitte unterstützen Sie dies, indem Sie defekte Teile nicht unnötig weiter zerstören.

Aufgabe 1: Logik-Tester

In dieser Aufgabe sollen die Eingabeplatine und die Logiktester auf der Ausgabeplatine kennengelernt werden. Dazu muss ein Versuchsaufbau entsprechend dem folgendem Foto und der Beschreibung dazu gemacht werden:



- Stromversorgung anschließen:
Eine GND-Verbindung (Null Volt, Minus, Masse) zwischen den Platinen mit einem schwarzen Kabeln stecken. Ebenso eine Verbindung zwischen +5V Pins mit einem roten Kabel herstellen. Außerdem das Netzteil farbrichtig (rot, schwarz) an einer Platine anschließen, aber noch nicht in die Steckdose stecken!
- Datenkabel anschließen:
Verbindung zwischen dem Taster Ta1 auf der Eingabeplatine mit dem Logiktestereingang TEST1 auf der Ausgabeplatine herstellen. Ebenso wird der Schalter S1 (im Foto S6) mit dem Logiktestereingang TEST2 verbunden. Hierbei dürfen keine schwarzen oder roten Kabel benutzt werden.
- Wenn der Aufbau damit dem Foto entspricht, dann darf das Netzteil in die Steckdose gesteckt werden.

Aufgaben:

Was wird bei der jeweiligen Schalterstellung angezeigt?

Was passiert, wenn der Taster gedrückt wird?

Was zeigt der Logiktester, wenn man das Verbindungskabel entfernt? (Dies ist eine Ausnahme - außer den Anschlüssen an den Logiktestern dürfen keine Kabel während des Betriebs umgesteckt werden.)

Was wird angezeigt, wenn man einen Logiktester mit +5V oder GND verbindet?

Hinweise:

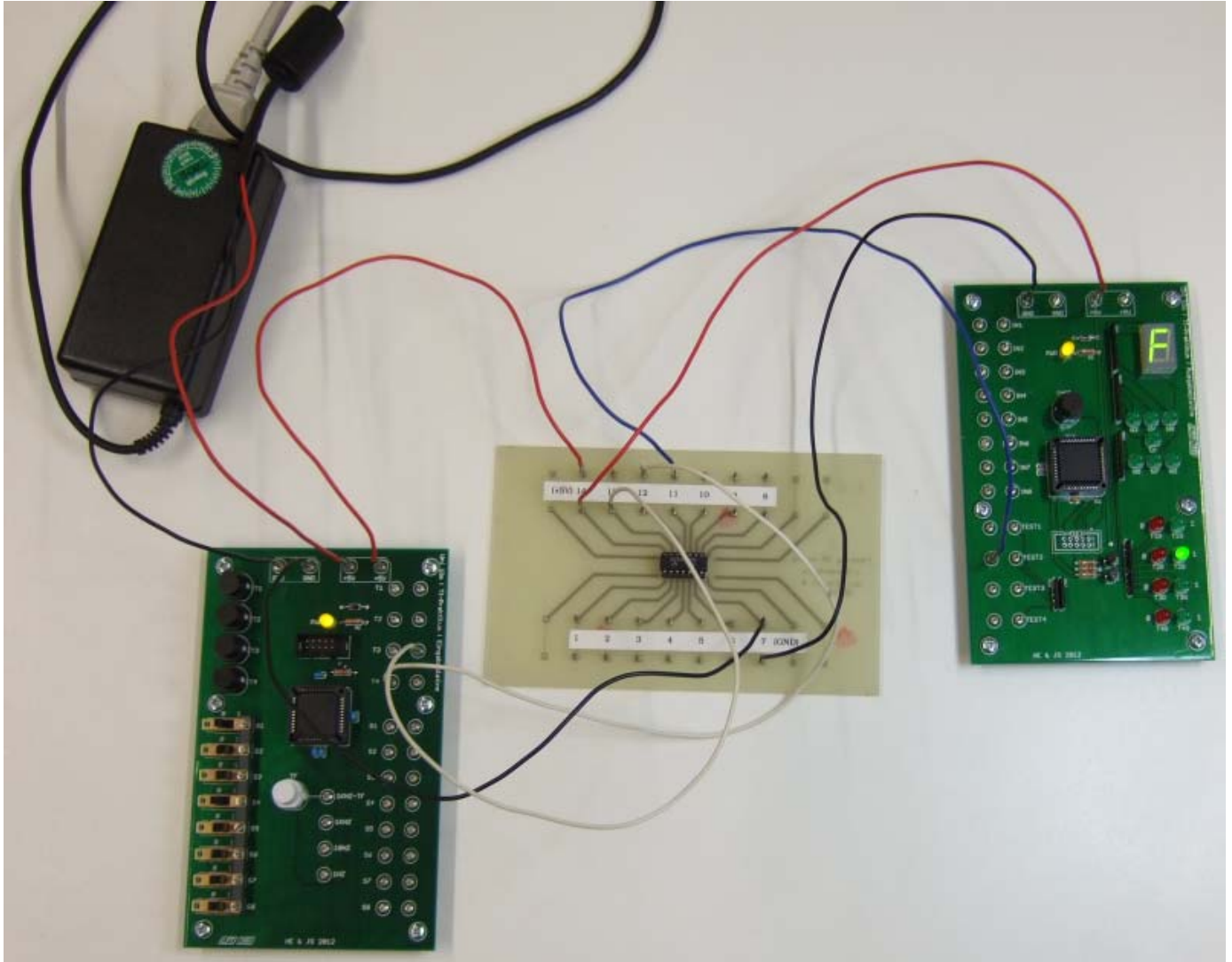
- Die Logiktester können später sowohl als Anzeigen für Resultate dienen, als auch zur Fehlersuche benutzt werden, indem Zwischenergebnisse überprüft werden (durch Anschließen im laufenden Betrieb).
- Der Zustand „offener Eingang“ wird nur von den Logik-Testern erkannt. Alle Eingänge von anderen Bausteinen im Praktikum müssen immer einen definierten Eingangswert bekommen.

Aufgabe 2: Untersuchung eines einfachen Gatters

In dieser Aufgabe soll ein Gatter mit zwei Eingängen als einfachster Fall einer booleschen Funktion untersucht werden.

- Der Betreuer gibt Ihnen einen Gattertyp vor. Wählen Sie ein geeignetes IC und stecken Sie es auf eine Leerplatine!
- Strom: Eingabeplatine, Leerplatine und Ausgabeplatine müssen mit roten und schwarzen Kabeln angeschlossen werden. Wenn Sie sich hier unsicher sind sollten Sie Ihren Betreuer fragen, da das Praktikumsmaterial hier am meisten gefährdet ist!
- Die Eingänge des Gatters mit Tastern, den Ausgang mit einem Logiktester verbinden.

Als Ergebnis müsste jetzt ein Aufbau entstanden sein, der dem folgenden Foto entspricht. Nur beim IC 74LS02 mit NOR-Gattern liegt das Gatter "andersherum" im IC, so dass die Verkabelung dann anders als im Foto sein muss.



Zuletzt den Strom einschalten, d.h., das Steckernetzteil in die Steckdose stecken. Wenn die Schaltung nicht das erwartete Verhalten zeigt sofort wieder ausschalten!

Aufgabe 3: Implementierung einer booleschen Funktion

In dieser Aufgabe soll eine vom Betreuer vorgegebene boolesche Funktion mit Logikbausteinen aufgebaut werden.

- Stellen Sie zuerst die Wahrheitstabelle für die Funktion theoretisch auf!
- Bauen sie eine Schaltung auf, die diese Funktion implementiert und überprüfen Sie diese! (Dieser Teil entfällt, wenn Sie gleich eine Schaltung entwerfen, die nur 2 ICs benötigt.)
- Bauen Sie nochmals eine Schaltung für diese Funktion auf. Diesmal dürfen jedoch nur 2 ICs verwendet werden, die nur Gatter enthalten dürfen (also keine Demultiplexer, Multiplexer, o.ä.). Überprüfen Sie auch diese Schaltung!

Hinweise:

- Die gegebene boolesche Funktion ist bereits optimal, d.h. ein Optimieren mittels Karnaugh-Diagramm erübrigt sich.
- Das Hauptproblem bei dieser Aufgabe ist, in Teil c) mit nur zwei ICs auszukommen. Formen Sie daher die gegebene boolesche Funktion geeignet um. Neben "normalen" Umformungsoperationen sind hier die DeMorganschen Regeln wichtig, um ODER-Operationen in UND-Operationen umzuformen (oder auch umgekehrt). Überprüfen Sie, ob ihre resultierende Funktion wirklich nur zwei ICs benötigt (sind genügend Inverter da?), bevor Sie diese realisieren. Welche ICs geeignet sind, ist ihre "Designerfreiheit".

Aufgabe 4: Ansteuern einer 7-Segment-Anzeige

In dieser Aufgabe soll die Funktion eines BCD-zu-7-Segment-Dekoders erarbeitet werden. Dazu soll eine Schaltung entworfen werden, um eines der Segmente der Anzeige zu steuern.

Um diese Aufgabe durchführen zu können, muß auf der Ausgabeplatine der jeweils passende Anzeigemodus mit dem Taster ausgewählt sein. Details hierfür stehen in der Beschreibung dieser Platine in der allgemeinen Praktikumseinleitung.

Zuerst stellt man Verbindungen zwischen den vier Tastern auf der Eingabeplatine und den Eingängen (IN1 bis IN4) auf der Ausgabeplatine her. Überprüfen Sie die korrekte Umsetzung der 4-Bit BCD-Zahl in eine Dezimal-Ziffer auf der Anzeige. Die Zahlen 0 bis 9 müssen korrekt angezeigt werden!

Es soll nun für ein vom Betreuer vorgegebenes Segment eine Ansteuerung entworfen werden. Nach Umstellen des Anzeigemodus kann dieses Segment über den Eingang IN8 direkt angesteuert werden.

Stellen Sie für dieses Segment eine Wahrheitstafel auf. Minimieren Sie die Funktion mit einem Karnaugh-Diagramm. Zeichnen Sie einen Schaltplan und bauen Sie die Schaltung auf. (Es sind wieder mehr als zwei ICs zulässig.)

Hinweis: Entwerfen Sie Ihre Schaltung zweimal. Beim ersten Mal dekodieren Sie die Einsen im Karnaugh-Diagramm. Beim zweiten Mal dekodieren Sie die Nullen und invertieren die daraus abgeleitete Funktion. Manchmal ist diese Funktion einfacher als die erste hergeleitete und deshalb auch einfacher zu bauen!

Hinweis 2: BCD-Zahlen haben die Kodierungen 0000 bis 1001, entsprechend 0 bis 9. Die Binärkodierungen 1010 bis 1111 müssen nicht in die Bilder von Hex-Ziffern A bis F umgesetzt werden. Nutzen Sie do-not-cares für diese Kodierungen um Ihre Schaltung zu vereinfachen.

Aufgabe 5: 1-Bit-Volladdierer

In dieser Aufgabe wird eine einfache Rechengrundfunktion implementiert. Die Recheneinheit soll die Eingänge Summand a, Summand b und den Übertrag c_{in} einer vorigen Stufe berücksichtigen. Als Ausgabe soll die Summe s und der neue Übertrag c_{out} berechnet werden.

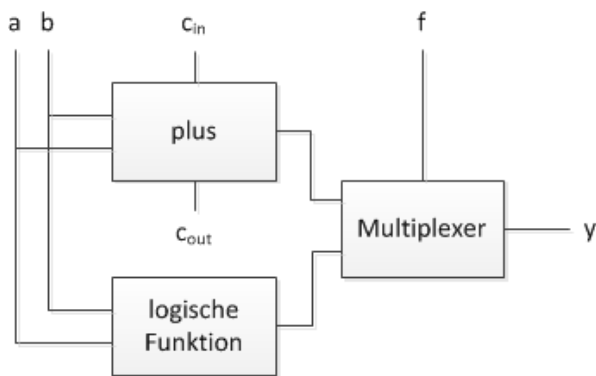
Achtung: der Versuchsaufbau wird für die folgende Aufgabe noch benötigt und sollte daher nicht gleich wieder abgebaut werden.

Stellen Sie die vollständige Wahrheitstafel für den Volladdierer auf. Minimieren Sie die Schaltfunktionen mittels Karnaugh-Diagrammen und zeichnen Sie ein Schaltbild, wobei nicht mehr als drei ICs verwendet werden sollen. (Sonst reicht es nicht mehr für die nächste Aufgabe.) Bauen Sie die Schaltung auf.

Für die Abnahme des Versuchs überlegen Sie sich bitte (nicht aufbauen!) wie ein 4-Bit-Volladdierer aufgebaut sein muss. Zeichnen Sie ein Blockschaltbild!

Aufgabe 6: Aufbau einer ALU mit zwei Funktionen

Der 1-Bit-Volladdierer soll nun zu einer ALU mit zwei Funktionen erweitert werden, entsprechend dem folgenden Blockschaltbild:



Die erste Funktion ist die Addition aus Aufgabe 5, deren Aufbau in dieser Aufgabe erweitert wird. Die zweite Funktion ist eine einfache boolesche Operation, wie z.B. NAND, und wird vom Betreuer vorgegeben. Mit dem Eingang f wird die Funktion der ALU ausgewählt. Bei $f=0$ soll die Addition durchgeführt werden, bei $f=1$ die andere Funktion. Als Dateneingänge dienen a, b und c_{in} , wobei die boolesche Funktion nur a und b berücksichtigt. Das Ergebnis wird an y und c_{out} ausgegeben. Bei der logischen Funktion liegt das Ergebnis an y an, und c_{out} darf eine beliebige Funktion sein.

Zeichnen Sie einen vollständigen Schaltplan und bauen Sie die Schaltung unter Verwendung des 1-Bit-Volladdierers aus Aufgabe 5 auf. Die Funktionswahl f soll über einen der Schalter (nicht Taster) erfolgen.

Entwerfen Sie eine Schaltung für den Multiplexer (realisierbar mit einem IC vom Typ 74LS00). Der Volladdierer existiert ja bereits und die logische Funktion ist trivial. Verschalten Sie die einzelnen Teile miteinander.

Hinweis: Alternativ zum obigen modularen Entwurf könnte man auch eine Wahrheitstabelle für die beiden ALU-Ausgänge in Abhängigkeit der vier Eingänge aufstellen und diese dann direkt mit Karnaugh-Diagrammen optimieren und aufbauen. Dies führt logischerweise zu einer effizienter arbeitenden ALU. Allerdings ist diese nicht-modulare Entwurfstechnik bei noch komplexeren Schaltungen nicht mehr praktikabel.

Aufgabe 7: Integrierte Multiplexer/Demultiplexer

In dieser Aufgabe werden fertig integrierte Multiplexer und Demultiplexer (DMPX, Dekodierer) verwendet. Diese besitzen sogenannte Enable-Eingänge, mittels denen die Grundfunktionalität eingeschaltet (Engl: Enable) werden muß. Beim Multiplexer (74LS151) muß Pin 7 auf 0 (Eingang ist invertiert) geschaltet werden, da der Multiplexerausgang sonst immer 0 liefern würde. Entsprechend müssen beim DMPX (74LS138) die Pins 4, 5 und 6 mit 0, 0 und 1 beschaltet werden, da sonst an allen Ausgängen immer 1 ausgegeben würde.

- Bauen Sie eine ALU mit vier logischen Funktionen auf. Die Funktionen werden vom Betreuer vorgegeben. Verwenden Sie einen Multiplexerbaustein (74LS151). Zeichnen Sie zuerst ein Prinzipschaltbild, anschließend ein detailliertes. Bauen Sie dann die Schaltung auf.
- theoretische Aufgabe:
Wie müßte eine 4-Bit Alu mit den Funktionen (Addition zweier 4-Bit-Zahlen), (UND), (EXOR) und (Null, unabhängig von den Eingängen) aussehen? Berücksichtigen Sie c_{in} und c_{out} . Zeichnen Sie ein Blockschaltbild! Mit welcher einfachen logischen Schaltung könnte die ALU um ein Zero-Flag ergänzt werden? (Ein Ausgang, der anzeigt, ob das Ergebnis der ALU den Wert Null hat.) Erweitern Sie ihr Blockschaltbild entsprechend!
- Steuern Sie dasselbe Segment der 7-Segmentanzeige an, das Ihnen in Aufgabe 4 vorgegeben wurde. Verwenden Sie den 1-zu-8-Demultiplexer 74LS138 und einige zusätzliche Gatter. Es sollen nur die Ziffern 0 bis 7 dekodiert bzw. dargestellt werden. Zeichnen Sie die Schaltung und bauen Sie diese auf. (Die Beschränkung auf die Ziffern von 0 bis 7 liegt am IC, das nur für diese Ausgänge besitzt. Demultiplexer mit mehr Ausgängen gibt es, jedoch sind diese nicht im Praktikumsmaterial enthalten.)

Hinweise: typischerweise benutzt man einen Dekodierer hier so, daß die Binärzahl diesen ansteuert. Die Ausgänge des Dekodierers werden dann mit Gattern zusammengefaßt, um das gewünschte Ergebnis zu erzeugen. Zu beachten ist außerdem, daß die Demultiplexerausgänge aktiv-low sind, d.h. nur der selektierte Ausgang ist log. Null, alle anderen sind log. Eins.

Wahrheitstabelle des 74LS138

| E1 Pin 4 | E2 Pin 5 | E3 Pin 6 | A0 Pin 1 | A1 Pin 2 | A2 Pin 3 | O0 Pin 15 | O1 Pin 14 | O2 Pin 13 | O3 Pin 12 | O4 Pin 11 | O5 Pin 10 | O6 Pin 9 | O7 Pin 7 |
|-------------|-------------|-------------|-------------|-------------|-------------|--------------|--------------|--------------|--------------|--------------|--------------|-------------|-------------|
| 1 | X | X | X | X | X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| X | 1 | X | X | X | X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| X | X | 0 | X | X | X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

Funktion des 74LS151

Die Funktion des Chips 74LS151 ist hier ausnahmsweise verbal statt mit Wahrheitstabelle beschrieben, da die Wahrheitstabelle recht gross, die verbale Beschreibung jedoch sehr einfach ist:

- Der 74LS151 ist ein 8-zu-1 Multiplexer. Der Wert an einem der Eingänge (Pins 4,3,2,1,15,14,13,12) wird auf den Ausgang Pin 5 durchgeschaltet, und zwar der, der durch die dreistellige Binärzahl an den Pins 9,10,11, entsprechend den Wertigkeiten 4, 2 und 1, ausgewählt ist. Beispiel: Pin 9 ist high, 10 und 11 low: der Wert von Pin 15 wird auf Pin 5 durchgeschaltet. Die Beschaltung an 9,10,11 ergibt zusammen den Wert 4 und Pin 15 ist die vierte oben genannte Zahl, weil die erste die "nullte" Zahl ist.
- Aber wenn der Enable-Eingang Pin 7 auf 1 ist, dann ist der Ausgang an Pin 5 immer 1.
- Der Ausgang an Pin 6 ist immer invertiert zum Ausgang an Pin 5.

Grundlagen der Rechnerarchitektur - Labor

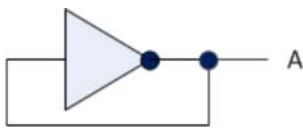
Versuch 2: Sequentielle Logik

Eine kleine Einführung in sequentielle Logik

Bei kombinatorischer Logik sind die Ausgangswerte eindeutig durch die Werte an den Eingängen definiert. Sequentielle Logik führt zusätzlich Speicherelemente ein, sogenannte Flip-Flops oder kurz FF, die jeweils genau ein Bit speichern können. Dies ermöglicht so triviale Dinge, wie einen Taster, der bei wiederholtem Drücken eine Lampe abwechselnd ein- und ausschaltet. Kombinatorische Logik alleine ist dazu nicht in der Lage.

Flip-Flops sind wieder aus den logischen Grundbausteinen aufgebaut, allerdings nutzen sie Rückkopplungen, d.h., ein Signal kann auf sich selbst einwirken. Rückkopplungen sind allerdings nicht beliebig zulässig. Hierzu zwei Gedankenexperimente.

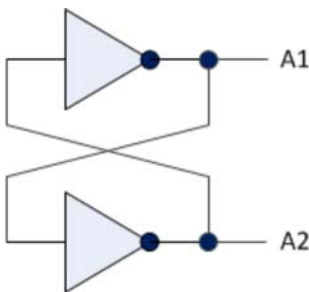
Gedankenexperiment 1



Wie verhält sich diese Schaltung? Ersichtlich ist die boolesche Gleichung $A=A'$ implementiert, die offensichtlich keine Lösung besitzt. Das Verhalten der Schaltung kann also von der Logik her nicht beantwortet werden.

In der Praxis muß diese Schaltung irgendein Verhalten haben. Hierbei kommen dann außer dem logischen Verhalten des Gatters dessen physikalische Eigenschaften zum Tragen. Die wichtigsten sind hier die zeitliche Verzögerung und die Verstärkung des Gatters. Derartige technische Details sind aber nicht Thema dieses Praktikums. Vielmehr bleiben wir hier im Bereich der Logik und halten hier nur fest, daß wir nicht jede Art von Rückkopplung zulassen können.

Gedankenexperiment 2



Nimmt man bei dieser Schaltung an, daß am Eingang des oberen Inverters (Signal A2) eine 1 liegt, so gibt der Inverter eine 0 an A1 aus. Der untere Inverter macht aus dieser 0 dann eine 1 für A2: der Kreis schließt sich, der Zustand ist stabil.

Hätte man zuerst eine 0 für A2 angenommen, so hätte man einen zweiten stabilen Zustand gefunden. Zwischen zwei Zuständen unterscheiden zu können ist aber genau die Informationsmenge, die man ein Bit nennt. Die Schaltung kann also genau ein Bit stabil speichern.

Mit boolescher Algebra ergibt sich das gleiche Ergebnis: die Inverter implementieren $A1=A2'$ und $A2=A1'$. Setzt man diese beiden Gleichungen ineinander ein, so erhält man $A1=A1$. Diese Gleichung besitzt zwei Lösungen: 0 und 1, also wieder die zwei stabilen Zustände.

Wie man sieht, ist hier die Rückkopplung mit der Logik der Gatter alleine zu erklären. Manche Rückkopplungen können also problemlos zugelassen werden, ohne daß Entwürfe instabil werden. Speziell sind Rückkopplungen obiger Art die Basis für Flip-Flops.

Nachteilig an der obigen Schaltung ist jedoch, daß man weder wählen kann, welchen Zustand die Schaltung annimmt, noch daß man diesen Zustand ändern könnte: diese Nachteile werden behoben im RS-FF, das in der ersten Aufgabe dieses Versuchs behandelt wird.

Technische Hinweise

Die Unterlagen für die in diesem Versuch benutzten ICs sind in den Unterlagen des ersten Versuchs mit enthalten, da sie (fast) nur mit kombinatorischen ICs zusammen sinnvoll genutzt werden können. Für diesen zweiten Versuch sollten Sie also die Anleitung des ersten Versuchs auch zur Hand haben.

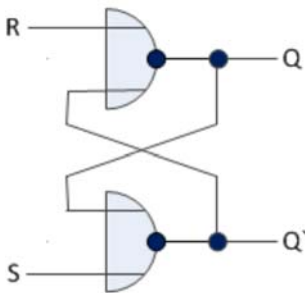
Ab der zweiten Aufgabe dieses Versuchs können Taktflanken auslösende Elemente sein, z.B. für einen Zähler, der genau Taktflanken zählen soll. In diesem Fall darf kein Eingabeelement benutzt werden, das "prellen" kann. "Prellen" bedeutet, dass eine Taste oder ein Schalter beim Betätigen kurz mehrere Umschaltvorgänge macht, typisch, weil die Metallteile, die den Kontakt im mikroskopischen Bereich herstellen, mehrfach Kontakt herstellen und wieder öffnen, ehe der neue Zustand stabil Kontakt bzw. Isolation ergibt. Im Praktikum sind alle Schalter und Taster elektronisch entprellt, und geben typisch wirklich nur einen Flanke pro Betätigung ab.

Oft benötigt man aber auch mehrere Tastendrucke hintereinander. Hier erleichtern Oszillatoren die Arbeit: man wählt eine passende Frequenz und benutzt diese für den Takt. Eine geeignete Frequenz ist im Praktikum meist eine, bei der man die Schaltvorgänge noch einzeln mitverfolgen kann: etwa ein Sekundentakt, also 1 Hz. Diese Frequenz liefert bei der Eingabeplatine der Pin "1 Hz".

Für die Anzeige von bis zu vier Binärsignalen kann die 7-Segment-Anzeige genutzt werden. Die Anzeige kann eine BCD-Ziffer darstellen, also für die Eingangswerte von 0000 bis 1001 die Ziffern 0 bis 9, für 1010 bis 1111 die Buchstaben A bis F. Es empfiehlt sich daher, bei bis zu vier Signalen die Eingänge IN1...IN4 in dieser Reihenfolge zu benutzen, wobei übrige mit 0 zu beschalten sind. Bleibt beispielsweise IN4 ungenutzt, so wird die Anzeige die Ziffern 0...7 darstellen, was den Binärwerten 000...111 entspricht. Natürlich muß die Ausgabeplatine im passenden Modus sein, damit die 7-Segment-Anzeige arbeitet.

Aufgabe 1: Das RS-Flip-Flop

In dieser Aufgabe soll das folgende RS-FF untersucht werden:



- Stellen Sie theoretisch die Wahrheitstabelle dafür auf. Eingangsgrößen sind R und S; Ausgangsgröße ist Q_{n+1} . Erklären Sie das FF!
- Bauen Sie das RS-FF auf und überprüfen Sie ihre Wahrheitstafel.

Hinweis:

Q_{n+1} und Q_n bezeichnen jeweils den Zustand von Q. Q_{n+1} ist der Zustand, der durch einen Schaltvorgang entsteht; Q_n der Wert, der vorher da war, also die Bildung von Q_{n+1} evtl. mitbestimmt.

Aufgabe 2: Untersuchen eines gegebenen FFs

Hier wird vom Betreuer die Schaltung eines Flip-Flops vorgegeben, dessen Eigenschaften zu untersuchen sind.

- Bauen Sie das FF auf und erstellen Sie eine Wahrheitstabelle. Eingänge sind hier A, B und C; Ausgänge sind HILF, X und Y (evtl. hat das Ihnen zugeteilte FF nicht alle diese Signale).
- Versuchen Sie das interne Verhalten des FFs zu verstehen, indem Sie sich dieses überlegen und/oder ausprobieren. Als Ergebnis dieses Aufgabenteils sollten Sie in der Lage sein, ihrem Betreuer das FF zu erklären.

Aufgabe 3: Digitalzähler

In dieser Aufgabe soll ein 2-Bit-Binärzähler aufgebaut werden. Benutzen Sie ein IC 74LS74 mit 2 D-FFs darin.

- Als Anzeige soll die 7-Segmentanzeige benutzt werden, die bei korrektem Zähleraufbau von 0 bis 3 zählen muß. (Schalten Sie die Eingänge IN3 und IN4 der Anzeige auf Masse, damit dies funktioniert.) Zeichnen Sie einen vollständigen Zyklus vom Verlauf des Eingangstaktes und den Ausgangspegeln an den beiden Q-Ausgängen auf!
- Ändern Sie ihren Zähler derart, daß er binär rückwärts zählt.

Aufgabe 4: Schieberegister

In dieser Aufgabe soll ein 4-Bit-Schieberegister aufgebaut werden. Benutzen Sie hierfür beide ICs 74LS74 bzw. 74LS107, die dem Experimentiermaterial beiliegen. Außer den beiden ICs mit den Flip-Flops ist maximal noch ein Inverter nötig!

- Bauen Sie ein solches Schieberegister auf. Als Anzeige sollen die vier Ecken der Würfelanzeige benutzt werden (IN2..IN5). Geben Sie den Verlauf der Ausgangspegel der Q-Ausgänge an, wenn am Eingang des Schieberegisters nacheinander die Werte 0, 0, 1, 0, 1, 1, 0, 0 angelegt werden.
- Erweitern Sie Ihren Zähler zum Möbius-Zähler, d.h., schalten Sie das invertierte Ausgangssignal des letzten FFs an den Eingang des ersten. Außerdem muß ein Möbiuszähler normalerweise mit dem Wert Null beginnen. Wie erreicht man dies? (Mit den neuen Chipsimulator-Platinen scheint das gar kein Problem zu sein. Nehmen Sie hier an, dass die FFs beim einschalten zufällige Werte annehmen können, nicht immer 0.) Geben Sie außerdem die Ausgangspegel der Q-Ausgänge für die ersten 9 Takte an!

Aufgabe 5: Würfel mit 7-Segment-Anzeige

In dieser Aufgabe soll ein Würfel aufgebaut werden, der die 7-Segment-Anzeige benutzt. Als Zähler ist ein IC 74LS161 zu benutzen. Die Daten des ICs finden sie auf der vorletzten Seite dieses Skriptes!

Entwickeln Sie eine Verschaltung für das IC 74LS161, die dieses immer wieder binär von 1 bis 6 zählen läßt. Wählen Sie auf der Ausgabeplatine die 7-Segment-Anzeige aus und verbinden Sie diese mit ihrem Zähler. Überprüfen Sie ihre Schaltung zuerst mit einzelnen Taktimpulsen, bevor Sie den Oszillator an den 1kHz Ausgang anschließen, der nur auf Tastendruck diese Frequenz liefert.

Zusatzfrage: Nach dem Einschalten kann das Zähler-IC zufällig auch in einem Zustand sein, der nicht zum Zählzyklus gehört (z.B. 0 oder 7). Wie bringt man den Zähler dann in den korrekten Zählzyklus? (Die entworfene Schaltung nicht ändern, nur überlegen!)

Aufgabe 6: Elektronischer Würfel als Zähler-Dekoder-Steuerung

Es soll nochmals ein elektronischer Würfel entwickelt werden, diesmal aber mit LEDs, die angeordnet sind wie die Augen eines normalen Würfels. Hierzu sind zwei Schritte nötig:

- Ein Zähler muß so beschaltet werden, daß er nur sechs Zustände durchläuft.
- Die sechs Zählerstände müssen so mittels Gattern dekodiert werden, daß mit den LEDs der Anzeigeplatine die bekannten Würfelbilder entstehen.

Für den ersten Schritt gibt der Betreuer ein Zähler-IC vor, von dem Sie dann auch die technischen Unterlagen erhalten. Außerdem gibt der Betreuer einen Zählbereich vor, wie z.B. 2-3-4-5-6-7. Ihre Aufgabe ist es das gegebene Zähler-IC so zu beschalten, dass es den vorgegebenen Zyklus durchläuft.

Um dies zu erreichen muss man den letzten gewünschten Zählerstand dekodieren, also im Beispiel oben die 7 bzw. je nach IC evtl. auch den danach, also die 8. Bei Erkennung des Zählerstands muss man den Zähler in den Anfangszustand des gewünschten Zählzyklus bringen, also im obigen Beispiel in den Zustand 2.

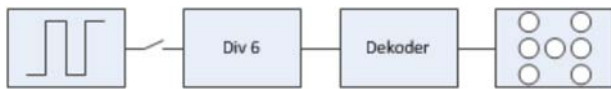
Entwerfen Sie eine Schaltung, bauen Sie diese auf und lassen Sie diese von Ihrem Betreuer abnehmen, ehe Sie weitermachen.

Für den zweiten Schritt entwerfen Sie einen Dekoder mit Logik-Gattern. Wie kombinatorische Logik entworfen wird sollte inzwischen bekannt sein. Sie dürfen natürlich freie Gatter aus dem ersten Schritt noch mitbenutzen.

Takten Sie Ihren Würfel zuerst mit einem Taster, ob genau alle sechs Würfelbilder pro Durchlauf jeweils genau ein mal entstehen. Funktioniert dies, so kann statt dem Taster der 1kHz Ausgang mit Taster der Eingabeplatine benutzt werden, so daß ein echter Würfel entsteht.

Eine Kurzbeschreibung des Würfels nochmals: Ein Taktgenerator läßt auf Tastendruck (= einmal würfeln) einen Zähler sehr schnell zählen. Sobald die Taste losgelassen wird, bleibt der zuletzt erreichte Zustand erhalten. Die sechs Zählerzustände werden so dekodiert, daß sieben Leuchtdioden ein Würfelbild anzeigen. Während des Zählens werden die Würfelzahlen so schnell durchlaufen, daß die Anzeige vom menschlichen Auge nicht mehr aufgelöst werden kann.

Als Blockschaltbild sieht das ganze wie folgt aus:

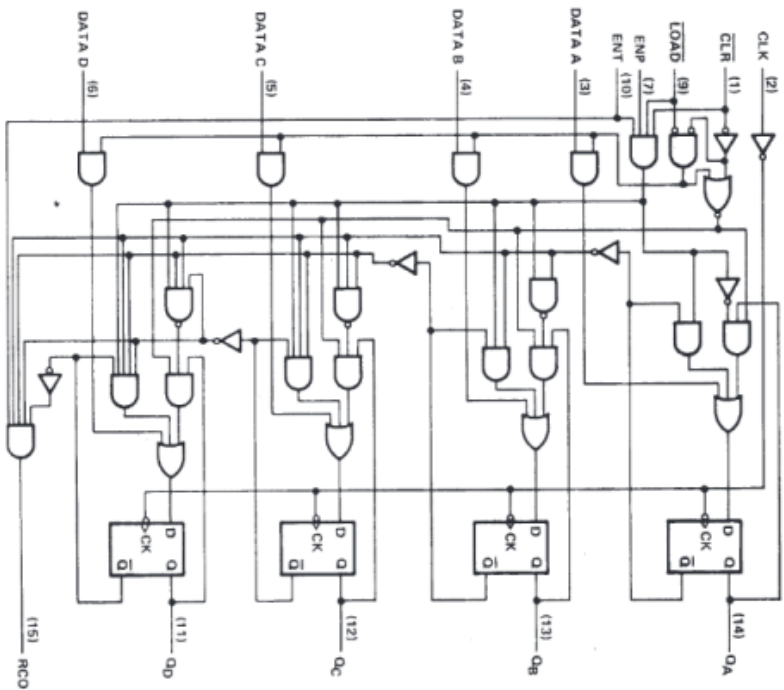


Überlegen Sie auch, wie schnell der Taktgenerator wenigstens sein muß, damit das menschliche Auge die Würfelbilder nicht mehr Auflösen kann. Geben Sie einen begründeten Schätzwert an!

logic diagram (positive logic)

SN54LS163A, SN74LS163A SYNCHRONOUS
BINARY COUNTERS

SN54LS161A, SN74LS161A synchronous binary
counters are similar; however, the clear is
asynchronous as shown for the SN54LS160A,
SN74LS160A decade counters at left.

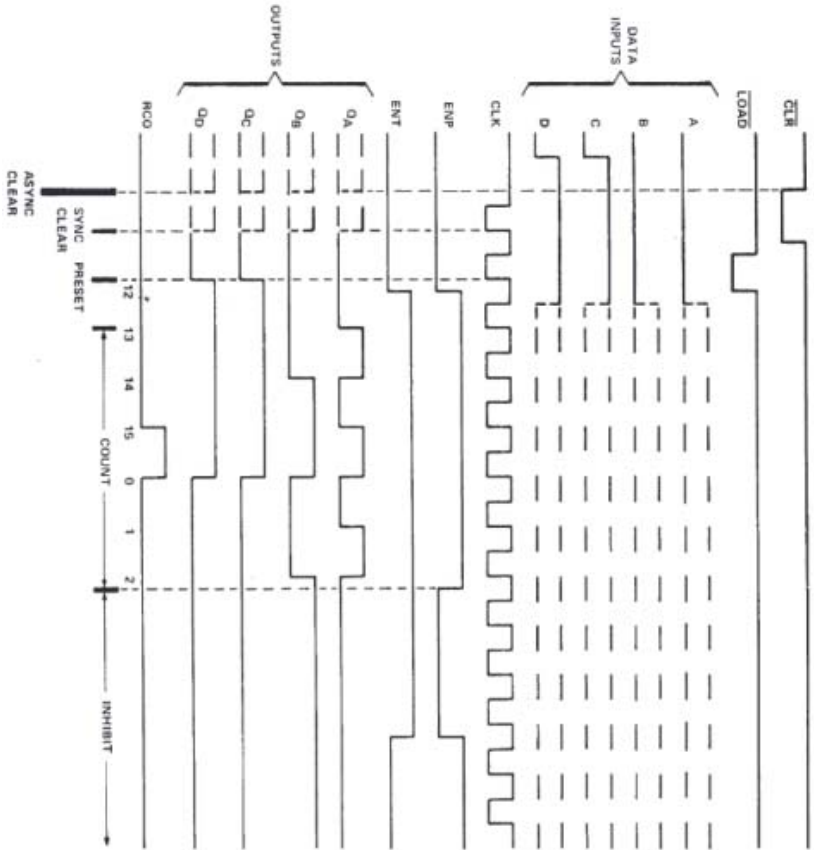


typical clear, preset, count, and inhibit sequences

'161, 'LS161A, '163, 'LS163A, 'S163 BINARY COUNTERS

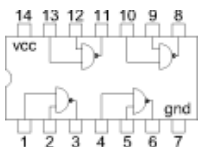
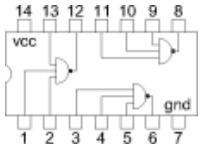
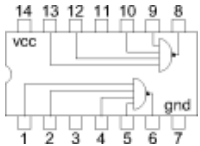
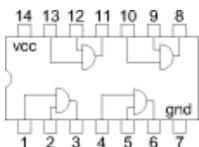
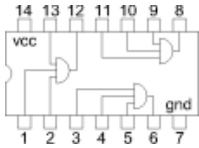
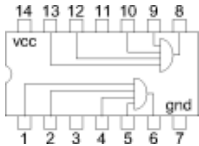
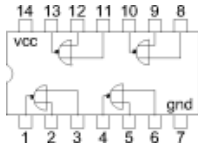
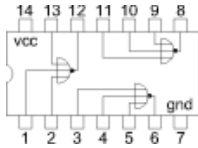
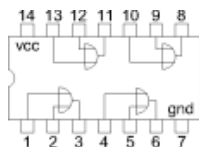
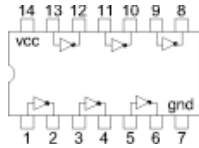
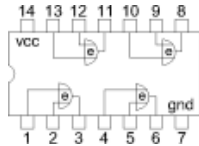
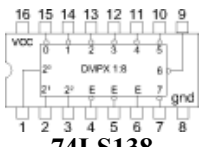
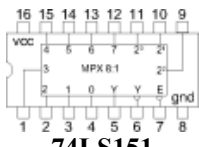
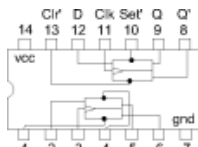
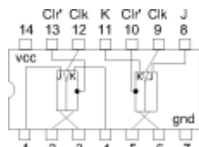
Illustrated below is the following sequence:

1. Clear outputs to zero ('161 and 'LS161A are asynchronous; '163, 'LS163A, and 'S163 are synchronous)
2. Preset to binary twelve
3. Count to thirteen, fourteen fifteen, zero, one, and two
4. Inhibit



Übersicht über die im Praktikum verwendeten ICs

Die Zahlen in Klammern geben an, welche Einstellung (S1, S2) bei einer Chipsimulatorplatine dieses IC ergibt.

| <p>NAND</p>  <p>74LS00 (2,2)</p>  <p>74LS10 (2,3)</p>  <p>74LS13 oder 74LS20 (2,4)</p> | <p>AND</p>  <p>74LS08 (0,2)</p>  <p>74LS11 (0,3)</p>  <p>74LS21 (0,4)</p> | <p>NOR</p>  <p>74LS02 (3,2, aber Gatter liegen wie beim 74LS00!)</p>  <p>74LS27 (3,3)</p> | <p>Sonstige</p>  <p>74LS32 (1,2)</p>  <p>74LS04 (5,1)</p>  <p>74LS86 (5,2)</p> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--|---|---|------------------|------------------|---|------------------|------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|-------|-------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|-----|---|----------------|----------------|--|-----|-----|---|---|------------------|------------------|---|---|---|---|---|---|---|---|---|---|----------------|----------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|----------------|----------------|---|-----|---|---|----------------|----------------|
|  <p>74LS138 (7,0)</p> |  <p>74LS151 (7,1)</p> |  <p>74LS74 (6,0)</p> <table><tr><th>Set</th><th>Clr</th><th>Clk</th><th>D</th><th>Q_{n+1}</th><th>Q_{n+1}</th></tr><tr><td>0</td><td>1</td><td>x</td><td>x</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>x</td><td>x</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>x</td><td>x</td><td>1 (B)</td><td>1 (B)</td></tr><tr><td>1</td><td>1</td><td>↑</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>↑</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>(A)</td><td>x</td><td>Q_n</td><td>Q_n</td></tr></table> <p>(A) 0 oder 1-Pegel sowie fallende Flanke, also alles außer steigender Flanke (B) die 1 ist nicht garantiert, d.h. nachfolgende Bausteine arbeiten evtl. nicht korrekt</p> | Set | Clr | Clk | D | Q _{n+1} | Q _{n+1} | 0 | 1 | x | x | 1 | 0 | 1 | 0 | x | x | 0 | 1 | 0 | 0 | x | x | 1 (B) | 1 (B) | 1 | 1 | ↑ | 1 | 1 | 0 | 1 | 1 | ↑ | 0 | 0 | 1 | 1 | 1 | (A) | x | Q _n | Q _n |  <p>74LS107 (6,1)</p> <table><tr><th>Clr</th><th>Clk</th><th>J</th><th>K</th><th>Q_{n+1}</th><th>Q_{n+1}</th></tr><tr><td>0</td><td>x</td><td>x</td><td>x</td><td>0</td><td>1</td></tr><tr><td>1</td><td>↓</td><td>0</td><td>0</td><td>Q_n</td><td>Q_n</td></tr><tr><td>1</td><td>↓</td><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>↓</td><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>↓</td><td>1</td><td>1</td><td>Q_n</td><td>Q_n</td></tr><tr><td>1</td><td>(A)</td><td>x</td><td>x</td><td>Q_n</td><td>Q_n</td></tr></table> <p>(A) 0 oder 1-Pegel sowie steigende Flanke, also alles außer fallender Flanke</p> | Clr | Clk | J | K | Q _{n+1} | Q _{n+1} | 0 | x | x | x | 0 | 1 | 1 | ↓ | 0 | 0 | Q _n | Q _n | 1 | ↓ | 1 | 0 | 1 | 0 | 1 | ↓ | 0 | 1 | 0 | 1 | 1 | ↓ | 1 | 1 | Q _n | Q _n | 1 | (A) | x | x | Q _n | Q _n |
| Set | Clr | Clk | D | Q _{n+1} | Q _{n+1} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | x | x | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | x | x | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | x | x | 1 (B) | 1 (B) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | ↑ | 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | ↑ | 0 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | (A) | x | Q _n | Q _n | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Clr | Clk | J | K | Q _{n+1} | Q _{n+1} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | x | x | x | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | ↓ | 0 | 0 | Q _n | Q _n | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | ↓ | 1 | 0 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | ↓ | 0 | 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | ↓ | 1 | 1 | Q _n | Q _n | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | (A) | x | x | Q _n | Q _n | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |