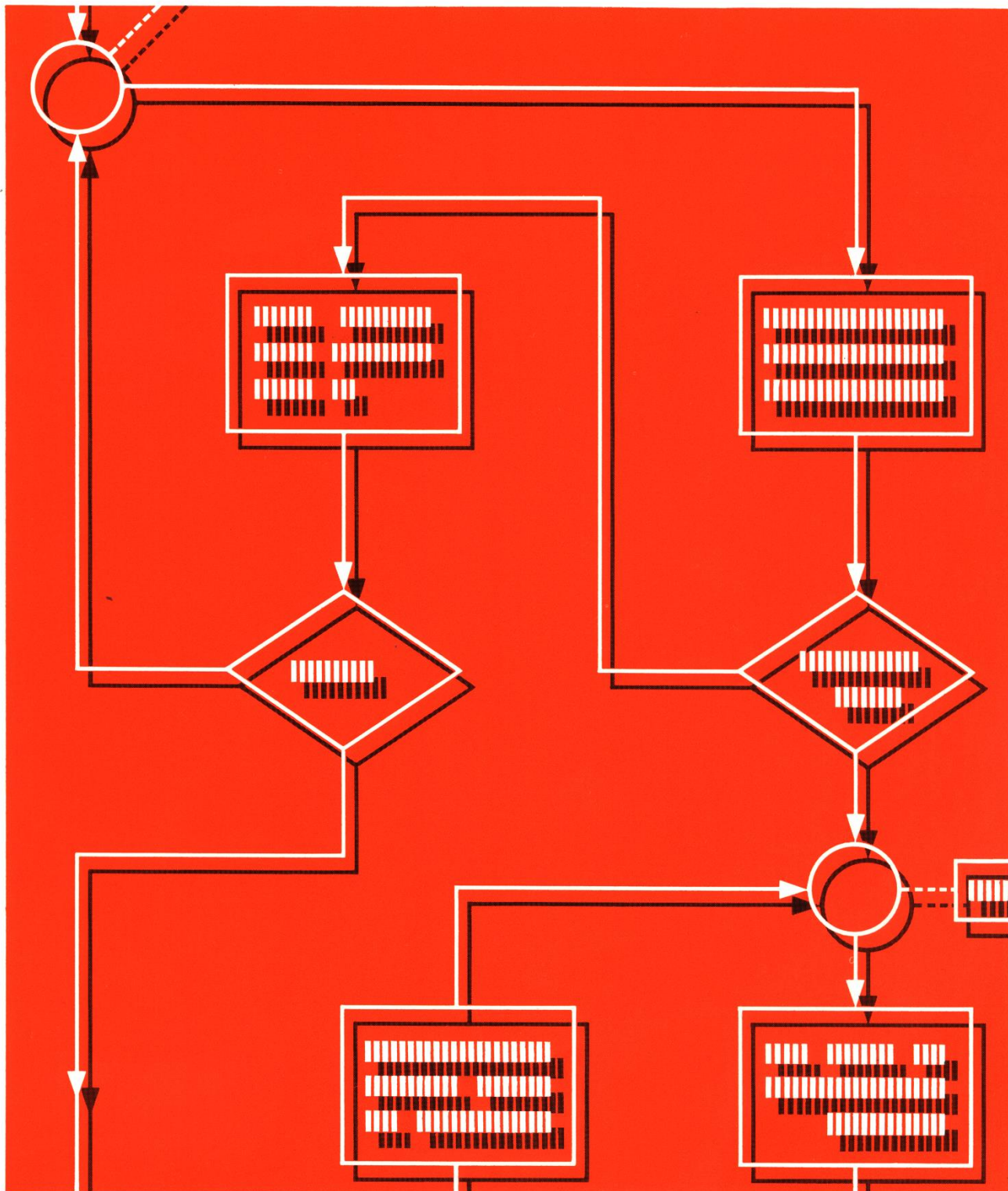


Mikroprozessor-Labor

Übungen



Übungen

Was ist beim Anfertigen und Einsenden der Lösungen der Prüfungsaufgaben zu beachten?

Es ist Ihnen freigestellt, ob Sie uns die Lösungen der Prüfungsaufgaben, die Sie auf der Seite Ü 3 vorfinden, zur Korrektur einsenden oder nicht. Bedenken Sie aber bitte: Unsere Korrekturen helfen Ihnen, Lücken in Ihrem Wissen zu erkennen. Unsere Noten geben Ihnen Aufschluß über den Erfolg Ihrer Arbeit.

Zur Einsendung sind nur die Lösungen der Prüfungsaufgaben bestimmt, die Sie auf den Seiten mit der Griffmarke Ü – in diesem Lehrbrief ab Seite Ü 3 – vorfinden. Die in den Abschnitten gestellten Aufgaben, zu denen wir Ihnen jeweils im selben Lehrbrief die Lösungen angeben, dienen nur Ihrer Selbstkontrolle. Ihre Lösungen dieser Übungsaufgaben sollen Sie nicht an uns einschicken.

Für die Aufgabenlösungen, die Sie uns zur Korrektur einsenden, haben wir diesem Lehrbrief Aufgabenlösungsblätter beigelegt. Verwenden Sie bitte nach Ihrem Belieben eine der beiden Seiten der Blätter.

Beschreiben Sie aber bitte grundsätzlich jedes Blatt nur auf einer Seite! Der Papierverbrauch wird dadurch zwar doppelt so groß. Wir nehmen das aber gern in Kauf, weil wir damit die Arbeit des Korrektors erleichtern.

Ihre Studiennummer und Ihr Name sollen auf jedes Blatt geschrieben werden, damit es keine Verzögerungen und Falschsendungen gibt. Ihre volle Anschrift braucht nur auf dem ersten Lösungsblatt zu stehen.

Einen Gutschein für weitere Aufgabenlösungsblätter haben wir beigelegt. Füllen Sie den Gutschein bitte aus und senden Sie ihn mit den ersten Aufgabenlösungen an uns ein. Wir schicken Ihnen die bestellten Blätter kostenlos und portofrei zu. Der Sendung liegt dann ein weiterer Gutschein für Ihre nächste Bestellung bei usw., so daß Sie fortlaufend mit Lösungsblättern versorgt sind und keine hinzukaufen müssen.

Ferner legen wir Briefumschläge verschiedener Größen bei. Verwenden Sie zum Einsenden Ihrer Prüfungsarbeiten an uns einen dieser bereits mit unserer Anschrift versehenen Umschläge.

Die großen Umschläge werden auf dem Postweg leicht beschädigt, wenn sich nur wenige Blätter darin befinden. Wenige Blätter falten Sie besser und stecken sie in einen kleineren Umschlag.

Legen Sie bitte Ihrer Sendung einen gleichgroßen Umschlag bei und vergessen Sie nicht, auf diesen Ihre Anschrift zu schreiben. In diesem Umschlag erhalten Sie Ihre Arbeiten zurückgesandt. Wenn Sie mit Ihren Arbeiten einen Gutschein zum Bezug von Aufgabenlösungsblättern einsenden, so legen Sie auf jeden Fall den mit Ihrer Anschrift versehenen großen Umschlag bei.

Die für Ihre erste Korrektursendung nicht benötigten Briefumschläge bewahren Sie bitte sorgfältig auf, falls Sie zum Einsenden der Aufgabenlösungen späterer Lehrbriefe andersformatige Umschläge benötigen.

Wir legen Ihnen bei jeder Rücksendung wieder einen gleichgroßen an uns adressierten Umschlag und einen weiteren bei, auf den Sie bitte wieder Ihre Anschrift schreiben wollen. Auf diese Weise haben Sie bei jeder Sendung an uns die Möglichkeit, das Umschlagformat der jeweiligen Anzahl der Blätter anzupassen.

Frankieren Sie bitte den an uns gerichteten Brief.

Das Porto für die Rücksendung tragen wir. Sie brauchen also den Umschlag, den Sie – mit Ihrer Adresse versehen – Ihrer Sendung an uns beilegen, nicht zu frankieren.

Sie können die Lösungen mehrerer Lehrbriefe zusammen an uns einsenden. Besonders bei den ersten Lehrbriefen ist es jedoch zweckmäßig, wenn wir Sie recht bald auf Denkfehler aufmerksam machen können. Wir empfehlen Ihnen deshalb, bei den ersten Einsendungen jeweils nur die Lösung eines Lehrbriefs einzuschicken.

Die Aufgaben sind in der gleichen Reihenfolge zu bringen, wie sie im Lehrbrief stehen! Bei Rechenaufgaben ist der ganze Rechnungsgang anzugeben und das Endergebnis zu unterstreichen. Für Unterstreichungen aber bitte keinen Rotstift benutzen, weil Rot die Korrekturfarbe ist! Zwischen den einzelnen Lösungen lassen Sie bitte Platz frei (1 cm Abstand genügt).

Achten Sie auf die vollständige Lösung und Einsendung aller Aufgaben. Lückenhafte Arbeiten müssen unkorrigiert zurückgesandt werden.

Sie können nur dann ein Abschlußzeugnis über die erfolgreiche Teilnahme am Lehrgang erhalten, wenn sämtliche Aufgabenlösungen zur Korrektur vorgelegt wurden.

Die einzelnen Blätter sind mit Hilfe von Briefklammern zusammenzuheften, wobei – bei der Einsendung mehrerer Lösungen – die niedrigste Lehrbriefnummer oben liegen soll. Wenn Sie die Lösungen mehrerer Lehrbriefe zugleich einsenden, stecken Sie bitte an das oberste Blatt der Sendung einen kleinen Zettel, auf den Sie beispielsweise schreiben: „Lösungen zu den Lehrbriefen 4 bis 6“.

Im übrigen vermeiden Sie bitte, kleine Zettel beizulegen. Auch Mitteilungen auf Postabschnitten usw. sowie Fragen, die zwischen die Aufgabenlösungen eingestreut sind, werden leicht übersehen. Für Fragen, die sich auf den Lehrstoff beziehen, legen Sie bitte ein besonderes Blatt A4 bei, das Sie mit Namen und Studiennummer versehen.

Die Rücksendung der korrigierten Lösungen mit dem nächstfolgenden Lehrbrief ist mit Rücksicht auf die Arbeitseinteilung am Institut nicht möglich; der Versand der Lehrbriefe erfolgt unabhängig von den korrigierten Lösungen.

Sehen Sie die korrigierten Lösungen genau durch! Finden Sie eine unklare Korrektur, so senden Sie diese mit einem entsprechenden Vermerk und sämtlichen zu diesem Lehrbrief gehörenden Lösungsblättern mit den folgenden Lösungen nochmals mit ein! Wir sind Ihnen dankbar, wenn Sie uns auf ein Versehen aufmerksam machen.

Die Einsendung der Aufgabenlösungen muß innerhalb von 5 Jahren erfolgen, vom Beginn des Studiums an gerechnet.

Wenn Sie diese Hinweise beachten, ist es uns möglich, Ihre Arbeiten in kürzester Zeit durchzusehen und zurückzusenden.

Prüfungsaufgaben Lehrbrief 1

Beachten Sie bitte beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü 1 und Ü 2.

1. In dem im Bild S 6.1 dargestellten Programmablaufplan wurde die Kämm-Routine ohne Rücksicht darauf eingeführt, ob das Kämmen jeweils notwendig ist oder nicht. Diese Verbeugung vor der Eitelkeit haben wir bereits in der Aufgabe S 6.1 kritisiert und dementsprechend nach der Anweisung „Verabschieden“ eine Verzweigung in das Hauptprogramm eingefügt, die in Abhängigkeit vom Ergebnis des Blicks in den Spiegel die Kämm-Routine aufruft oder den Kamm unbenutzt läßt (vgl. Bild Ü 5.1).

Sicher wäre es sinnvoll, die Anweisung „Spiegelgucken“ und nachfolgend die Verzweigung mit „Frisur in Ordnung?“ vor jeder Kämm-Routine in das Hauptprogramm einzufügen.

Programmtechnisch eleganter ist es, die **jedesmal** vor der Kämm-Routine erscheinende Anweisung „Spiegel gucken“ zu einem Teil des Unterprogramms zu machen. Das Unterprogramm heißt dann nicht mehr „Kämmen“, sondern „Kämmen wenn nötig“.

Skizzieren Sie das Hauptprogramm entsprechend dem Bild S 6.1 mit dieser Änderung! – Beachten Sie, daß eine Anweisung aus dem Bild S 6.1 jetzt wegfallen kann!

Skizzieren Sie – ähnlich wie im Bild S 10.1 – das neue Unterprogramm neben dem Hauptprogramm! Gehen Sie dabei von der Darstellung im Bild S 7.1 aus und ergänzen Sie diese Darstellung durch die Anweisung „Spiegel gucken“ und die nachfolgende Verzweigung mit „Frisur in Ordnung?“ – Fügen Sie vor der Grenzstelle „Ende Kämmen“ eine Anweisung „zurück zum Hauptprogramm“ ein! (Vgl. Seite S 10.) – Überlegen Sie genau, wohin die Wege nach der Verzweigung mit „Frisur in Ordnung?“ führen!

Skizzieren Sie die Sinnbilder für die Operationen in der Form und mit dem Seitenverhältnis, wie es die Bilder im Abschnitt „Der Programmablaufplan“ zeigen!

2. Im Versuch H 39.1 haben Sie festgestellt, daß sich die Äquivalenz-Verknüpfung von zwei Eingangssignalen aus der Exklusiv-ODER-Verknüpfung (Antivalenz-Verknüpfung, vgl. Seite H 38) ergibt, wenn man das Ausgangssignal dieser Verknüpfung anschließend negiert.

Skizzieren Sie diesen Zusammenhang unter Verwendung der Schaltzeichen für Exklusiv-ODER-Verknüpfung, Äquivalenz-Verknüpfung und NICHT-Glied so, wie wir es im Bild H 36.2 für die NOR- und die NAND-Verknüpfung gezeigt haben.

3. Wenden Sie auf die folgenden Gleichungen die de Morganschen Theoreme an:

a) $a = \overline{e_1 \wedge e_2 \wedge e_3}$

b) $a = \overline{\overline{e_1} \vee e_2}$

c) $a = e_1 \wedge \overline{e_2}$

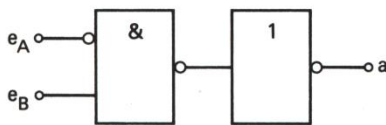


Bild Ü 4.1

Zu Prüfungsaufgabe 4a. Für die hier dargestellte Verknüpfung braucht man nur ein einziges Schaltzeichen.

4. a) Die im Bild Ü 4.1 dargestellte Verknüpfung der Signale e_A und e_B mit zwei Schaltgliedern kann unter Verwendung eines einzigen Schaltzeichens dargestellt werden. Skizzieren Sie dieses Schaltzeichen!
- b) Vervollständigen Sie die folgende verbale Formulierung der Funktion des entsprechend der Teilaufgabe a skizzierten Verknüpfungsglieds:
Das Ausgangssignal a hat dann (und nur dann) den Wert 1, wenn das Eingangssignal e_A den Wert...
- c) Schreiben Sie die entsprechende algebraische Gleichung an!
- d) Wenden Sie auf die von Ihnen gefundene algebraische Gleichung das entsprechende de Morgansche Theorem an!

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe H 2.1

Die Schaltung kann durch das Leuchten der Lumineszenzdiode melden, daß in einem Raum mit zwei Türen die Gefahr des Durchzugs besteht.

Am Ausgang des NAND-Glieds erscheint dann – und nur dann – ein 0-Signal, wenn am einen Eingang des NAND-Glieds UND (gleichzeitig) am anderen Eingang 1-Signale liegen. – Dieser Sachverhalt läßt sich auch so ausdrücken:

Am Ausgang des NAND-Glieds erscheint immer dann ein 1-Signal, wenn am einen Eingang ODER am anderen Eingang ein 0-Signal liegt.

Das 1-Signal am Ausgang wird von der gegen L-Potential geschalteten Lumineszenzdiode durch Leuchten gemeldet.

Wenn Ihnen das Lösen dieser Aufgabe Schwierigkeiten gemacht hat, dann lassen Sie sich bitte nicht entmutigen: Wir werden in einem der ersten Hardware-Abschnitte die notwendigen Grundlagen der Digitaltechnik kurz erläutern.

Zu Aufgabe S 6.1

Das Bild Ü 5.1 zeigt den geänderten Teil des Programms entsprechend dem Bild S 6.1.

Nach der Anweisung „Spiegel gucken“ folgt eine Verzweigung in Abhängigkeit von der Entscheidung „Frisur in Ordnung?“. Wenn die Frisur in Ordnung ist (Ja), dann kann das Kämmen unterbleiben. Es folgt sofort die Anweisung „Mantel anziehen“.

Fällt die Entscheidung jedoch so aus, daß die Frisur nicht in Ordnung ist (Nein), dann wird der Programmzweig mit dem Aufruf des Unterprogramms „Kämmen“ durchlaufen. Danach folgt die Zusammenführung mit dem Hauptprogramm, in dem die nächste Anweisung „Mantel anziehen“ lautet.

Zu Aufgabe S 8.1

Wenn nach dem Verlassen des Ankleidezimmers das Fähnchen nicht entfernt wird, geschieht zunächst gar nichts: Unser Professor marschiert programmgemäß zur Haustür.

Schlimm wird es jedoch am nächsten Morgen. Der Professor betritt zunächst das Ankleidezimmer, ohne sich um das Fähnchen zu kümmern. Beim Verlassen sieht er jedoch das Fähnchen vom vergangenen Morgen und entschreitet – wie programmiert – ohne Frühstück durch die Haustür. (Vorausgesetzt natürlich, daß er diesmal korrekt gekleidet ist. Wenn nicht, dann geht er wieder ins Ankleidezimmer, nimmt das Fähnchen aus der Öse und steckt es – laut Anweisung – wieder hinein. Er wird sich nicht mal wundern, denn er ist ja ein zerstreuter Professor.)

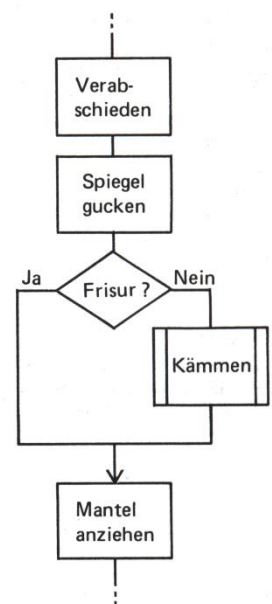


Bild Ü 5.1
Zu Aufgabe S 6.1. Das Unterprogramm „Kämmen“ wird nur dann aufgerufen, wenn bei der Verzweigung die Entscheidung „Frisur in Ordnung? Nein!“ gefällt wird.

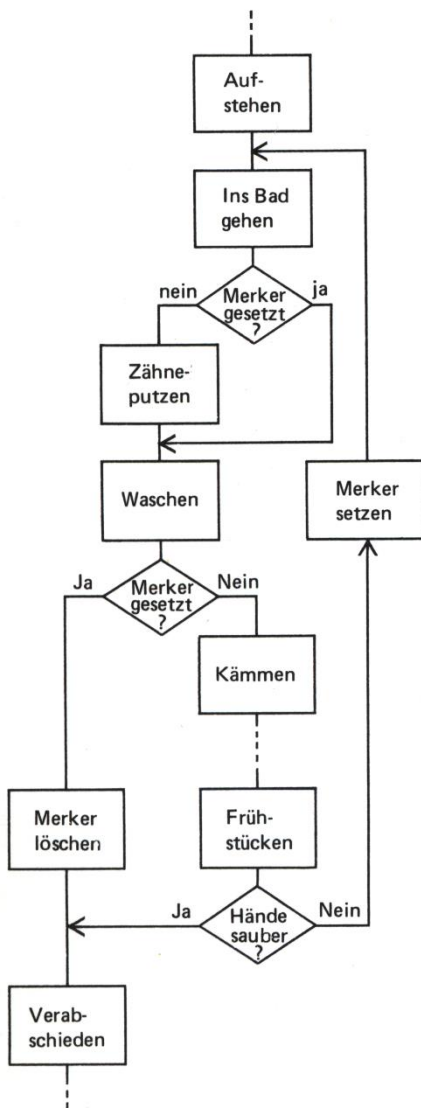


Bild Ü 6.1
Zu Aufgabe S 9.1. Der gleiche Merker kann innerhalb eines Programms auch mehrfach abgefragt werden.

Zu Aufgabe S 9.1

Der Programmablaufplan im Bild Ü 6.1 spricht für sich selbst. Beachten Sie, daß der Merker zweimal abgefragt wird: Zuerst zum Umgehen der Anweisung „Zähneputzen“ und noch einmal nach der Anweisung „Waschen“. Erst danach darf der Merker gelöscht werden.

Zu Aufgabe H 33.1

a) Bild Ü 6.2a

b) Es bieten sich zwei verbale Formulierungen an:

Die Ausgangsvariable a des NOR-Glieds hat immer dann den Wert 0, wenn die Eingangsvariable e_A ODER die Eingangsvariable e_B den Wert 1 haben.

Diese Formulierung entspricht der Formulierung beim ODER-Glied, mit dem Unterschied, daß man sich auf den Wert 0 der Ausgangsvariablen a bezieht.

Die zweite mögliche Formulierung entspricht der des UND-Glieds (vgl. Seite H 29):

Die Ausgangsvariable a des NOR-Glieds hat dann (und nur dann) den Wert 1, wenn die Eingangsvariable e_A UND (gleichzeitig) die Eingangsvariable e_B die Werte 0 haben.

Hier bezieht man sich – abweichend von der Formulierung beim UND-Glied – allerdings auf die Werte 0 der Eingangsvariablen.

c) Bild Ü 6.2b

b)

e_A	e_B	e_C	a
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

a)

e_A	e_B	a
0	0	1
0	1	0
1	0	0
1	1	0

Bild Ü 6.2
Zu Aufgabe H 33.1: Funktionstabellen für die NOR-Verknüpfung
a) Zwei Eingangsvariable,
b) drei Eingangsvariable.

Zu Aufgabe H 35.1

Die Umwandlung einer UND-Verknüpfung in eine NAND-Verknüpfung wird durch Negation des Ausgangssignals der UND-Verknüpfung erreicht. In den Funktionstabellen für die NAND-Verknüpfung im Bild Ü 7.1 erkennen Sie diese Negation im Vergleich mit den Funktionstabellen für die UND-Verknüpfung im Bild H 30.1.

a)

e_A	e_B	a
0	0	1
0	1	1
1	0	1
1	1	0

b)

e_A	e_B	e_C	a
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Bild Ü 7.1

Zu Aufgabe H 35.1: Funktionstabellen für die NAND-Verknüpfung.

a) Zwei Eingangsvariable,
b) drei Eingangsvariable.

Zu Aufgabe H 36.1

- a) Das Signal e wird zweimal negiert. Wenn am Eingang e das Signal $e = 0$ liegt, erscheint am Ausgang des ersten NICHT-Glieds ein 1-Signal. Die Negation dieses 1-Signals im zweiten NICHT-Glied ergibt das Signal $a = 0$ am Ausgang a .
- b) Nach der hier angestellten Überlegung hat das Signal am Ausgang a stets den gleichen Wert wie das Signal am Eingang e . Es ist also $a = e$

Bitte beachten Sie, daß wir sorgsam zwischen dem Anschluß eines Schaltglieds (Eingang,Ausgang) und dem an dem betreffenden Anschluß liegenden Signal unterscheiden!

Die Eingangsanschlüsse von Schaltgliedern bezeichnen wir mit gerade gedruckten Buchstaben e , die Ausgangsanschlüsse mit gerade gedruckten Buchstaben a .

Der Anschluß eines Schaltglieds ist z.B. ein lötbare Draht, sonst nichts. Wichtig ist, daß ein solcher Anschluß nicht negiert werden kann. Oder wissen Sie, was ein negierter Draht sein könnte?

Nur weil es übersichtlich ist, bezeichnen wir die variablen Signale, die an einem Anschluß erscheinen, mit dem gleichen Buchstaben und dem gleichen Index wie den Anschluß selbst. Die Tatsache, daß es sich um Variable handelt, machen wir durch den Druck deutlich:

Die Eingangsvariablen von Schaltgliedern bezeichnen wir mit *kursiv* gedruckten Buchstaben e , die Ausgangsvariablen mit *kursiv* gedruckten Buchstaben a .

Bei Schaltzeichen wird die Negation am Eingang oder am Ausgang nur durch den kleinen Kreis dargestellt, niemals durch einen Negations-Querstrich!

Zu Aufgabe H 38.1

Das Bild Ü 7.2 zeigt die Funktionstabelle der Exklusiv-ODER-Verknüpfung.

Zu Aufgabe H 39.1

Die Negation des Ausgangssignals der Exklusiv-ODER-Verknüpfung bewirkt, daß bei jeder Eingangssignal-Kombination der antivalente Wert des Ausgangssignals a erscheint, der in die Funktionstabelle im Bild Ü 7.2 eingetragen ist. Das Bild Ü 7.3 zeigt die neue Funktionstabelle.

e_A	e_B	a
0	0	0
0	1	1
1	0	1
1	1	0

Bild Ü 7.2

Zu Aufgabe H 38.1: Funktionstabelle der Exklusiv-ODER-Verknüpfung.

e_A	e_B	a
0	0	1
0	1	0
1	0	0
1	1	1

Bild Ü 7.3

Zu Aufgabe H 39.1: Funktionstabelle der Äquivalenz-Verknüpfung.

Zu Aufgabe H 43.1

a) $a = \overline{\overline{e_A} \vee \overline{e_B}}$

b) $a = \overline{\overline{e_A} \wedge \overline{e_B} \wedge \overline{e_C}}$

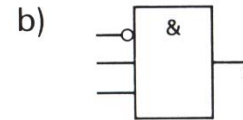
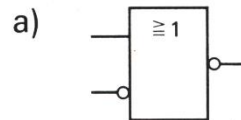
c) Bild Ü 8.1a

d) Bild Ü 8.1b

Bild Ü 8.1

Zu Aufgabe H 43.1c und d.

Aus dem UND-Glied mit einem negierenden Eingang wird ein NOR-Glied mit einem anderen negierenden Eingang (Teilbild a). Das NOR-Glied mit drei Eingängen, von denen zwei negieren, wird zum UND-Glied mit einem negierenden Eingang (Teilbild b).



Prüfungsaufgaben Lehrbrief 2

Bitte beachten Sie beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü 1 und Ü 2.

1. Mit dem ab der Seite S 20 beschriebenen Universal-Verknüpfungs-Programm soll die Exklusiv-ODER-Verknüpfung (vgl. ab Seite H 37) der Signale e_A und e_B nachgebildet werden.

- a) Tragen Sie in den Programmablaufplan entsprechend dem Bild S 21.1 auf dem Lösungsblatt an der richtigen Stelle die Anweisung
 $a = e_A \oplus e_B$
 ein.

Überlegen Sie, was mit den Anweisungen ②, ⑤ und ⑦ geschehen muß und tragen Sie gegebenenfalls die Anweisung „LEER“ ein.

- b) Schreiben Sie das Programm entsprechend der Tabelle im Bild S 21.2 an. Der Befehl XRE für die Exklusiv-ODER-Verknüpfung hat den Operationscode 60.

Wenn eine Negations-Anweisung aus dem Programm herausgenommen werden muß, dann wird jeder Teil des entsprechenden Negations-Doppelbefehls E4 04 durch einen Leerbefehl 08 ersetzt.
 Sie können Ihr Programm im Versuch prüfen! (Bild Ü 7.2).

2. Das Mikroprozessor-System soll mit dem ab der Seite S 20 beschriebenen Universal-Verknüpfungsprogramm die NOR-Verknüpfung

$$a = \overline{e_A \vee e_B}$$

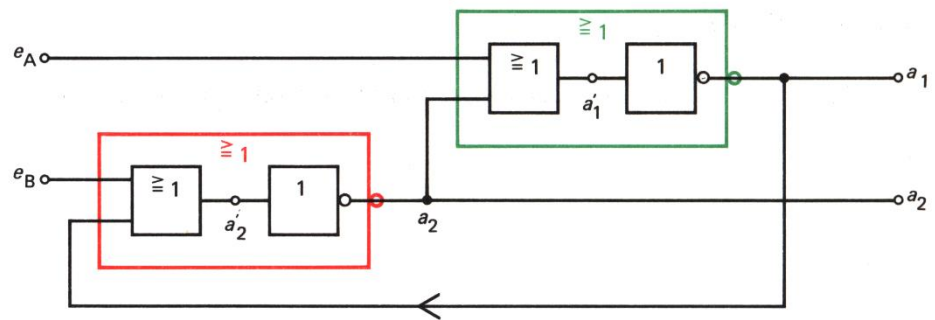
vornehmen.

Es soll der Programmablaufplan und das Programm für diese Aufgabe angegeben werden, wenn die Anweisung ⑥ (Bild S 21.1) einen UND-Verknüpfungsbefehl ANE mit dem Operationscode 50 enthält.

- a) Wenden Sie auf die Gleichung $a = \overline{e_A \vee e_B}$ das 1. De Morgansche Theorem (Seite H 41) an. (Vgl. Seite H 43!)
- b) Ergänzen Sie den für die unter a) gefundene Gleichung maßgeblichen Programmablaufplan auf dem Lösungsblatt entsprechend dem Bild S 21.1 und überlegen Sie, welche der Negationen ②, ⑤ und ⑦ durch LEER-Anweisungen ersetzt werden müssen. Tragen Sie an der richtigen Stelle die Anweisung „UND-Verknüpfung“ ein.
- c) Schreiben Sie das Programm entsprechend der Tabelle im Bild S 21.2 an und berücksichtigen Sie an den richtigen Stellen die Operationscodes 50 (UND-VERKNÜPFEN) und 08 (LEER).
- d) Tasten Sie das Programm ein, löschen Sie gegebenenfalls die Lumineszenzdioden (Versuch H 28.1) und starten Sie das Programm! Ergänzen Sie die Funktionstabelle auf dem Lösungsblatt!

Bild Ü 10.1

Zu Prüfungsaufgabe 3: Schaltplan eines NOR-Basisflipflops. Die Darstellung entspricht dem Schaltplan im Bild H 48.1. Wie muß das Programm im Bild H 48.2 geändert werden, damit das Mikroprozessor-System diese Schaltung nachbildet?



3. Im Bild Ü 10.1 ist der Schaltplan eines NOR-Basisflipflops dargestellt. Sein Verhalten kann mit dem Mikroprozessor-System studiert werden, wenn das Programm im Bild H 48.2 an vier Stellen geändert wird.
 - a) Studieren Sie den Programmablaufplan im Bild H 49.1. Geben Sie an, welche Anweisungen geändert werden müssen und wie diese Anweisungen nach der Änderung aussehen.
 - b) Geben Sie an, welche Befehle im Programmablaufplan (Bild H 48.2) geändert werden und wie diese Befehle nach der Änderung aussehen! (Leerbefehl: 08, ODER-Befehl: 58.)
 - c) Programmieren Sie das Mikroprozessor-System mit dem geänderten Programm aus dem Bild H 48.2!
Welche Ausgangssignal-Kombination von a_1 und a_2 stellt sich bei der irregulären Eingangssignal-Kombination $e_A = 1$; $e_B = 1$ ein?
4. Wandeln Sie die folgenden Zeichenfolgen in die Dezimaldarstellung um:
 - a) $A4_{16}$; b) $5C_{16}$; c) $2B6_{16}$; d) $A3B_{16}$; e) 543_{16} ; f) $A34C_{16}$
5. Stellen Sie die folgenden, dezimal dargestellten Zahlen sedezimal dar:
 - a) 19_{10} ; b) 23_{10} ; c) 426_{10} ; d) 1024_{10} ; e) 7648_{10} ; f) 37625_{10}
6. Ihre Quarzuhr mit Ziffernanzeige (Seite H 71) geht nach 24 Stunden um 9 s vor. Welche sedezimale Zeichenkombination müssen Sie unter der Nummer 04AB in das Mikroprozessor-System eingeben, wenn dort vorher $A2_{16}$ stand?

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe S 14.1

- a) Die Ausgangsvariable a des NAND-Glieds mit negierendem Eingang e_A hat dann (und nur dann) den Wert 0, wenn die Eingangsvariable e_A den Wert 0 hat UND (gleichzeitig) die Eingangsvariable e_B den Wert 1 hat.
- b) Bild Ü 11.1
- c) $\bar{a} = \bar{e}_A \wedge e_B$

e_A	e_B	a
0	0	1
0	1	0
1	0	1
1	1	1

Bild Ü 11.1

Zu Aufgabe S 14.1b: Funktionstabelle eines NAND-Glieds mit negierendem Eingang e_A .

Zu Aufgabe S 21.1

- a) Die Ausgangsvariable a des NAND-Glieds mit zwei negierenden Eingängen hat dann (und nur dann) den Wert 0, wenn die Eingangsvariable e_A den Wert 0 hat UND (gleichzeitig) die Eingangsvariable e_B den Wert 0 hat.
- b) Bild Ü 11.2
- c) $\bar{a} = \bar{e}_A \wedge \bar{e}_B$
- d) $\bar{a} = \overline{e_A \vee e_B}$; $a = e_A \vee e_B$
Ein NAND-Glied mit zwei negierenden Eingängen entspricht einem ODER-Glied.

e_A	e_B	a
0	0	0
0	1	1
1	0	1
1	1	1

Bild Ü 11.2

Zu Aufgabe S 21.1b: Funktionstabelle eines NAND-Glieds mit zwei negierenden Eingängen

Zu Aufgabe S 24.1

Wir setzen voraus, daß das Mikroprozessor-System das zum Versuch S 23.1 gehörende Programm enthält.

- a) Die Umprogrammierung wird nach folgendem Schema vorgenommen:

TASTEN	ANZEIGE	BEMERKUNGEN
RESET	0420 06	Das vielleicht noch laufende Programm wird angehalten. Die Befehlsnumerierung kehrt zur Anfangsnummer 0420 zurück.
4 3 1 AD	0431 08	Die zur Anweisung ⑦ gehörende Befehlsnummer (0)431 wird vorgewählt.
E 4	IN 0432 08	Die unter den Nummern 0431 und 0432 stehenden Leerbefehle 08 werden durch den Doppelbefehl E4 04 (INVERTIEREN) überschrieben. Durch IN wird die Befehlsnummer automatisch erhöht.
0 4	IN 0433 07	
(RESET	0420 06)	Diese Eingabe ist nicht notwendig, schadet aber nicht. Die Befehlsnumerierung kehrt zur Anfangsnummer 0420 zurück.
4 2 5 AD	0425 E4	Die zur Anweisung ② gehörende Befehlsnummer (0)425 wird vorgewählt.

e_A	e_B	a
0	0	1
0	1	1
1	0	0
1	1	1

Bild Ü 12

Zu Aufgabe S 24.1b: Funktionstabelle der NAND-Verknüpfung des Signals e_A mit dem negierten Signal e_B .

TASTEN	ANZEIGE	BEMERKUNGEN
0 8 IN 0426 04		Der erste Teil des Doppelbefehls E4 04 wird vom Leerbefehl 08 überschrieben. IN bewirkt die Erhöhung der Befehlsnummer auf 0426.
0 8 IN 0427 01		Der zweite Teil des Doppelbefehls E4 04 wird vom Leerbefehl 08 überschrieben. IN bewirkt die Erhöhung der Befehlsnummer auf 0427.
GO	verlischt	Das Programm wird bei der Anfangsnummer gestartet.

b) Bild Ü 12.1

Zu Aufgabe H 58.1

Sie können jetzt bereits feststellen, daß das Flipflop **nicht taktzustandsgesteuert** ist. Wäre es taktzustandsgesteuert, dann müßte das Flipflop während des Taktsignals $C = 1$ durch das Steuersignal gekippt werden können (Seite H 56).

Wir haben es also mit einem taktflankengesteuerten Flipflop zu tun. Ob es sich um eine 0-1- oder um eine 1-0-Flankensteuerung handelt, läßt sich noch nicht feststellen.

Zu Aufgabe S 29.1

a) 4 mal zweihundertsechsfünzig + 2 mal sechzehn + 0 mal eins oder:
 $4_{16} \cdot 16_{10}^2 + 2_{16} \cdot 16_{10}^1 + 0_{16} \cdot 16_{10}^0$ oder:
 $4_{10} \cdot 256_{10} + 2_{10} \cdot 16_{10} + 0_{10} \cdot 1 = 1056_{10}$
 $420_{16} = \text{eintausendsechsfünzig}$

b) 5 mal zweihundertsechsfünzig + A mal sechzehn + B mal eins oder:
 $5_{16} \cdot 16_{10}^2 + A_{16} \cdot 16_{10}^1 + B_{16} \cdot 16_{10}^0$ oder:
 $5_{10} \cdot 256_{10} + 10_{10} \cdot 16_{10} + 11_{10} \cdot 1 = 1451_{10}$
 $5AB_{16} = \text{eintausendvierhundertsechsfünzig}$

c) F mal sechzehn + 4 mal eins oder:
 $F_{16} \cdot 16_{10}^1 + 4_{16} \cdot 16_{10}^0$ oder:
 $15_{10} \cdot 16_{10} + 4_{10} \cdot 1_{10} = 244_{10}$
 $F4_{16} = \text{zweihundertvierundvierzig}$

d) 0 mal zweihundertsechsfünzig + 9 mal sechzehn + D mal eins oder:
 $0_{16} \cdot 16_{10}^2 + 9_{16} \cdot 16_{10}^1 + D_{16} \cdot 16_{10}^0$ oder:
 $0_{10} \cdot 256_{10} + 9_{10} \cdot 16_{10} + 13_{10} \cdot 1 = 157_{10}$
 $09D_{16} = \text{einhundertsiebenundfünzig}$

- e) F mal viertausendsechshundneunzig + A mal zweihundertsechshundfünfzig
 + B mal sechzehn + C mal eins
 $F_{16} \cdot 16^3_{10} + A_{16} \cdot 16^2_{10} + B \cdot 16^1_{10} + C \cdot 16^0_{10}$
 oder:
 $15_{10} \cdot 4096_{10} + 10_{10} \cdot 256_{10} + 11_{10} \cdot 16_{10} + 12_{10} \cdot 1_{10} = 64188_{10}$
 $FABC_{16}$ = vierundsechzigtausendeinhundertachtundachtzig.

Zu Aufgabe S 32.1

- a) In 166_{10} ist $16^2_{10} = 256_{10}$ nicht enthalten.
 $166_{10} : 16^1_{10} = 166_{10} : 16_{10} = 10_{10} = A_{16}$ (Rest 6_{10}) **A .**
 $6_{10} : 16^0_{10} = 6_{10} : 1_{10} = 6_{10} = 6_{16}$ **A 6**
 Einhundertsechshundsechzig = $A6_{16}$
- b) In 1147_{10} ist $16^3_{10} = 4096_{10}$ nicht enthalten
 $1147_{10} : 16^2_{10} = 1147_{10} : 256_{10} = 4_{10} = 4_{16}$ (Rest 123_{10}) **4 . .**
 $123_{10} : 16^1_{10} = 123_{10} : 16_{10} = 7_{10} = 7_{16}$ (Rest 11_{10}) **47 .**
 $11_{10} : 16^0_{10} = 11_{10} : 1_{10} = 11_{10} = B_{16}$ **47B**
 Eintausendeinhundertsiebenundvierzig = $47B_{16}$
- c) In 3021_{10} ist $16^3_{10} = 4096_{10}$ nicht enthalten
 $3021_{10} : 16^2_{10} = 3021_{10} : 256_{10} = 11_{10} = B_{16}$ (Rest 205_{10}) **B . .**
 $205_{10} : 16^1_{10} = 205_{10} : 16_{10} = 12_{10} = C_{16}$ (Rest 13_{10}) **BC .**
 $13_{10} : 16^0_{10} = 13_{10} : 1_{10} = 13_{10} = D_{16}$ **BCD**
 $3021_{10} = BCD_{16}$
- d) In 3850_{10} ist $16^3_{10} = 4096_{10}$ nicht enthalten.
 $3850_{10} : 16^2_{10} = 3850_{10} : 256_{10} = 15_{10} = F_{16}$ (Rest 10_{10}) **F . .**
 $10_{10} : 16^1_{10} = 10_{10} : 16_{10} = 0_{10} = 0_{16}$ (Rest 10_{10}) **F0 .**
 $10_{10} : 16^0_{10} = 10_{10} : 1_{10} = 10_{10} = A_{16}$ **F0A**
 $3850_{10} = F0A_{16}$
- e) In 51783_{10} ist $16^4_{10} = 65536_{10}$ nicht enthalten.
 $51783_{10} : 16^3_{10} = 51783_{10} : 4096_{10} = 12_{10} = C_{16}$ (Rest 2631_{10}) **C . . .**
 $2631_{10} : 16^2_{10} = 2631_{10} : 256_{10} = 10_{10} = A_{16}$ (Rest 71_{10}) **CA . .**
 $71_{10} : 16^1_{10} = 71_{10} : 16_{10} = 4_{10} = 4_{16}$ (Rest 7_{10}) **CA4 .**
 $7_{10} : 16^0_{10} = 7_{10} : 1_{10} = 7_{10} = 7_{16}$ **CA47**
 $51783_{10} = CA47_{16}$

Zu Aufgabe H 63.1

- a) Es können Zweihundertsechsfundfünfzig unterschiedliche Zahlen (einschließlich der Zahl Null) abgelegt werden.
- b) FF = Zweihundertfünfundfünfzig

Prüfungsaufgaben Lehrbrief 3

Beachten Sie bitte beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü 1 und Ü 2.

1. Bei dem von uns verwendeten Mikroprozessor wird die im Pointer-Register PC gespeicherte Befehlsnummer bei laufendem Programm jeweils unmittelbar vor der Ausführung des nächsten Befehls erhöht. (Wir werden uns mit dieser Tatsache noch eingehend beschäftigen.) Bei einem Programm-Sprung (Seite H 93) muß das Pointer-Register PC deshalb mit einer Befehlsnummer geladen werden, die um eins kleiner ist als die Nummer des Befehls, der als nächster ausgeführt werden soll.

Ein einfacher Sprungbefehl hat den Operationscode 90 (Seite H 93). Er bewirkt durch Addition oder Subtraktion direkt die Änderung des Inhalts von PC um die sedezimal dargestellte Zahl, die auf den Sprungbefehl 90 unter der nächstfolgenden Nummer abgelegt ist. Der Sprungbefehl mit dem Operationscode 90 ist also ein Zwei-Byte-Befehl (Seite H 88).

In irgendeinem Programm findet sich der folgende Befehl:

0438 90 } Sprungbefehl; der Inhalt des Pointer-Registers PC soll von
0439 57 } 0439₁₆ um 57₁₆ erhöht werden.

Unter welcher Befehlsnummer steht der nächste Befehl, der ausgeführt wird?

2. Ein Programm soll von der Befehlsnummer 057A₁₆ zur Befehlsnummer 05C5₁₆ springen. Es muß also gefragt werden: Wie groß ist der Abstand (die Differenz) zwischen 05C4₁₆ und 057A₁₆, denn unmittelbar vor der Befehlsausführung des unter der Nummer 05C5₁₆ abgelegten Befehls wird der Inhalt von PC noch einmal um eins erhöht. Wie groß ist also die Differenz zwischen 05C4₁₆ und 057A₁₆?

3. In irgendeinem Augenblick des Programmlaufs sind die Register des Mikroprozessors mit folgenden Bitmustern belegt:

(AC) = 0110 1110 (EX) = 1010 0011

(ST): CY/L = 1; OV = 0; SA = 0; SB = 0; IE = 1; FO = 1; F1 = 0; F2 = 1
(Bild H 86.1!)

(PC) = 0000 0100 0011 0001 (P2) = 0000 0000 1101 1001

(P1) = 0000 1000 0000 0110 (P3) = 1010 0101 1011 0111

a) Geben Sie die Inhalte sedezimal dargestellt an.

b) Geben Sie die Inhalte dezimal dargestellt an.

4. Geben Sie das Bitmuster an, mit dem die Zahl Eintausendvierhundertundzwanzig in einem der Pointer-Register abgelegt wird!
5. Führen Sie schriftlich folgende Subtraktionen durch Addition des Zweierkomplements aus:
- a) $10100011 - 1101110$; b) $11001010 - 1110010$
c) $100111001 - 11111011$

Schreiben Sie die Ergebnisse sedezimal dargestellt an.

6. Der Versuch H 93.1 (Blinkprogramm mit Lumineszenzdioden F2, F1 und F0) soll so abgewandelt werden, daß abwechselnd die Lumineszenzdioden F0 und F1 gemeinsam und die Lumineszenzdiode F2 allein leuchtet. Die Lumineszenzdiode F2 soll jeweils halb so lange leuchten wie die Lumineszenzdioden F0 und F1.

Das Programm hat die gleiche Länge wie das zum Versuch H 93.1. Auch der Aufbau ist der gleiche. Sie brauchen nur die Anweisungen ① und ④ zu ändern. Nehmen Sie die Änderungen ohne Rücksicht darauf vor, welche Lumineszenzdioden nach dem Start des Programms (GO) zuerst aufleuchten!

- a) Skizzieren Sie den vollständigen Programmablaufplan für das abgewandelte Programm und versehen Sie die Anweisungen mit Nummern (vgl. Bild H 93.1).
- b) Schreiben Sie sämtliche, zum abgewandelten Programm gehörenden Befehle zusammen mit den Befehlsnummern so an, wie wir es z.B. im Bild S 17.1 für das Programm zur NAND-Verknüpfung getan haben. Tragen Sie die Nummern für die Anweisungen und den zugehörigen Text entsprechend dem Programmablaufplan ein.

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Test gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe S 35.1

1. Summand:

$$D_{16} \cdot 16_{10}^2 + A_{16} \cdot 16_{10}^1 + 9_{16} \cdot 16_{10}^0 = 13_{10} \cdot 256_{10} + 10_{10} \cdot 16_{10} + 9_{10} \cdot 1_{10} \\ = 3497_{10}$$

2. Summand:

$$B_{16} \cdot 16_{10}^2 + 5_{16} \cdot 16_{10}^1 + 8_{16} \cdot 16_{10}^0 = 11_{10} \cdot 256_{10} + 5_{10} \cdot 16_{10} + 8_{10} \cdot 1_{10} \\ = 2904_{10}$$

Summe:

$$6401_{10}$$

In 6401_{10} ist $16_{10}^4 = 65536_{10}$ nicht enthalten.

$$6401_{10} : 16_{10}^3 = 6401_{10} : 4096_{10} = 1_{10} = 1_{16} \text{ (Rest } 2305_{10}) \quad 1 \dots$$

$$2305_{10} : 16_{10}^2 = 2305_{10} : 256_{10} = 9_{10} = 9_{16} \text{ (Rest } 1_{10}) \quad 19 \dots$$

$$1_{10} : 16_{10}^1 = 1_{10} : 16_{10} = 0_{10} = 0_{16} \text{ (Rest } 1_{10}) \quad 190 \dots$$

$$1_{10} : 16_{10}^0 = 1_{10} : 1_{10} = 1_{10} = 1_{16} \quad 1901$$

Ergebnis: $6401_{10} = 1901_{16}$

Diese Zeichenfolge haben wir auch bei der Addition der sedezimal dargestellten Zahlen erhalten.

Zu Aufgabe S 37.1

a) $7C_{16} + 3B_{16} = B7_{16}$

b) $135_{16} + 427_{16} = 55C_{16}$

c) $ABC_{16} + DEF_{16} = 18AB_{16}$

d) $1E4A_{16} + 2AC1_{16} = 490B_{16}$

Zu Aufgabe S 40.1

Der jeweils sedezimal dargestellte Minuend, der Subtrahend und die Differenz wird dezimal dargestellt.

Minuend:

C mal zweihundertsechsfünzig + 2 mal sechzehn + 3 mal eins oder:

$$\begin{array}{rcll} C_{16} \cdot & 16^2_{10} & + 2_{16} \cdot & 16^1_{10} + 3_{16} \cdot 16^0_{10} \text{ oder:} \\ 12_{10} \cdot & 256_{10} & + 2_{10} \cdot & 16_{10} + 3_{10} \cdot 1_{10} = 3107_{10} \end{array}$$

Subtrahend:

A mal zweihundertsechsfünzig + 2 mal sechzehn + 6 mal eins oder:

$$\begin{array}{rcll} A_{16} \cdot & 16^2_{10} & + 2_{16} \cdot & 16^1_{10} + 6_{16} \cdot 16^0_{10} \text{ oder:} \\ 10_{10} \cdot & 256_{10} & + 2_{10} \cdot & 16_{10} + 6_{10} \cdot 1_{10} = 2598_{10} \end{array}$$

Differenz:

1 mal zweihundertsechsfünzig + F mal sechzehn + D mal eins oder:

$$\begin{array}{rcll} 1_{16} \cdot & 16^2_{10} & + F_{16} \cdot & 16^1_{10} + D_{16} \cdot 16^0_{10} \text{ oder:} \\ 1_{10} \cdot & 256_{10} & + 15_{10} \cdot & 16_{10} + 13_{10} \cdot 1_{10} = 509_{10} \end{array}$$

In Dezimal-Darstellung ergibt sich: $3107_{10} - 2598_{10} = 509_{10}$

Zu Aufgabe S 40.2

$$\begin{array}{r} \text{a) } F \quad C_{16} \\ -3 \quad 2_{16} \\ \hline C \quad A_{16} \end{array}$$

$$\begin{array}{r} \text{b) } 4 \quad 6' \quad 7_{16} \\ \quad \quad 5_{16} \quad 17_{16} \\ -4 \quad 5 \quad A_{16} \\ \hline 0 \quad 0 \quad D_{16} \end{array}$$

$$\begin{array}{r} \text{c) } 5' \quad A' \quad 6_{16} \\ \quad \quad 4_{16} \quad 19_{16} \quad 16_{16} \\ -2 \quad C \quad 7_{16} \\ \hline 2 \quad D \quad F_{16} \end{array}$$

$$\begin{array}{r} \text{d) } F \quad F \quad 7 \quad C_{16} \\ \quad \quad \quad 6_{16} \quad 1C_{16} \\ -A \quad F \quad 5 \quad F_{16} \\ \hline 5 \quad 0 \quad 1 \quad D_{16} \end{array}$$

Zu Aufgabe S 42.1

$$\begin{array}{r} 4 \quad 2 \quad A \quad 3_{16} \\ + \quad 6 \quad 5 \quad F_{16} \\ + \quad 1 \quad 1 \quad 1 \quad \text{Übertrag} \\ \hline 4 \quad 9 \quad 0 \quad 2_{16} \end{array}$$

Zu Aufgabe S 43.1

Das Mikroprozessor-System liefert:

$$FE31_{16} - EF99_{16} = (0)E98_{16}$$

Schriftlich rechnen Sie nach folgendem Schema:

$$\begin{array}{r}
 \begin{array}{cccc}
 F & E & 3 & 1 \\
 E & 1D & 12 & 11
 \end{array} \\
 - E & F & 9 & 9 \\
 \hline
 (0) & E & 9 & 8
 \end{array}$$

Zu Aufgabe H 73.1

- a) Die größte Zahl ist dann dargestellt, wenn an allen vier Stellen das höchstwertige Sedezimalzeichen F erscheint:

$$FFFF_{16}$$

- b) F mal viertausendsechshundneunzig + F mal zweihundertsechshundfünfzig + F mal sechzehn + F mal eins

oder

$$F_{16} \cdot 16^3_{10} + F_{16} \cdot 16^2_{10} + F_{16} \cdot 16^1_{10} + F \cdot 16^0_{10}$$

oder

$$15_{10} \cdot 4096_{10} + 15_{10} \cdot 256_{10} + 15_{10} \cdot 16_{10} + 15_{10} \cdot 1_{10} = 65535_{10}$$

Zu Aufgabe S 47.1

Mit fünf Stellen ergibt sich die Zeichenkombination $1\ 1\ 1\ 1\ 1_2$ als Darstellung der größten Zahl im Dualsystem. Diese Zeichenkombination bedeutet:

$$\text{Einmal } 2^4_{10} + \text{einmal } 2^3_{10} + \text{einmal } 2^2_{10} + \text{einmal } 2^1_{10} + \text{einmal } 2^0_{10} = 31_{10}$$

Die größte Zahl, die mit fünf Stellen im Dualsystem dargestellt werden kann, ist also Einunddreißig.

Zu Aufgabe S 47.2

Wir geben nur für die Aufgabe a den Lösungsweg noch einmal an; er ist so einfach, daß wir ihn uns für die übrigen Aufgaben sparen können.

- a) Die am weitesten links stehende Ziffer hat den Stellenwert 2^3_{10} . Es ergibt sich also:

$$1 \cdot 2^3_{10} + 0 \cdot 2^2_{10} + 1 \cdot 2^1_{10} + 0 \cdot 2^0_{10} = 8_{10} + 0_{10} + 2_{10} + 0_{10} = 10_{10}$$

b) 21_{10} c) 49_{10}

d) Vorsicht! In der Aufgabe hat die Zeichenfolge 1 0 1 1 1 0 0 keine tiefgestellte Ziffer 2, die die Zeichenfolge als Dualdarstellung kennzeichnet! Streng genommen darf so etwas nicht vorkommen. Aber wo führte das hin, wenn man bei jeder Dezimaldarstellung einer Zahl eine Kennzeichnung durch tiefgestellte Zeichen 10 vornehmen wollte! Wir wollen uns also dahingehend einigen, daß die fehlende Kennzeichnung einer Zeichenfolge eine Dezimaldarstellung bedeuten soll.

Nur wenn es sich ganz eindeutig um eine im Mikroprozessor verarbeitete Zahl handelt, soll eine nicht gekennzeichnete Zeichenfolge eine Dezimaldarstellung bedeuten.

Übrigens: $1\ 0\ 1\ 1\ 1\ 0\ 0_2 = 92_{10}$

e) 197_{10}

In Ihrem Mikroprozessor-System entspricht eine leuchtende Lumineszenzdiode der Ziffer 1, eine nicht leuchtende Lumineszenzdiode der Ziffer 0. – Vergleichen Sie die Lösung der Aufgabe e mit dem auf der Seite H 68 gezeigten Muster und der Zahl, die dieses Muster darstellen soll!



Zu Aufgabe S 48.1

a) 111_2 b) 1101_2 c) 100100_2 d) 110110100_2
e) 11110111001_2

Zu Aufgabe S 48.2

Nebenstehendes Bild. Kennen Sie die LOLLO?

Zu Aufgabe S 50.1

Wir geben Ihnen hier nur die Ergebnisse der Umwandlungen an. Die Rechengänge haben Sie bei den Lösungen der vorhergehenden Aufgaben geübt.

1. Beispiel: $110_2 = 6_{10}$; $11_2 = 3_{10}$; $3_{10} + 6_{10} = 9_{10}$; $9_{10} = 1001_2$

2. Beispiel: $1011_2 = 11_{10}$; $111_2 = 7_{10}$; $11_{10} + 7_{10} = 18_{10}$; $18_{10} = 10010_2$

Zu Aufgabe S 50.2

Auch bei den Lösungen zu dieser Aufgabe verzichten wir auf die Angaben der Rechengänge:

a) $26_{10} = 11010_2$; $11_{10} = 1011_2$;

$26_{10} + 11_{10} = 37_{10}$

$$\begin{array}{r} 11010 \\ + 1011 \\ \hline 100101 \end{array}$$

$100101_2 = 37_{10}$

b) $173_{10} = 10101101_2$; $231_{10} = 11100111_2$

$173_{10} + 231_{10} = 404_{10}$

$$\begin{array}{r} 10101101 \\ + 11100111 \\ \hline 110010100 \end{array}$$

$110010100_2 = 404_{10}$

c) $1369_{10} = 10101011001_2$; $507_{10} = 11111011_2$;

$1369_{10} + 507_{10} = 1876_{10}$

$$\begin{array}{r} 10101011001 \\ + 11111011 \\ \hline 11101010100 \end{array}$$

$11101010100_2 = 1876_{10}$

Zu Aufgabe S 53.1

Das Komplement ergibt sich aus $10000 - 76 = 9924$. Jetzt folgt:

$$\begin{array}{r} 1923 \\ + 9924 \\ \hline 11847 \end{array}$$

Von der in der Summe ganz links stehenden Ziffer wird eins abgezogen.
Das Ergebnis 1847 stimmt mit dem Ergebnis der Subtraktion überein.

Zu Aufgabe S 53.2

Das Komplement finden Sie durch folgende Subtraktion

$(16^2_{10} = 256_{10} = 100_{16}):$

$$\begin{array}{r} 100_{16} \\ - 3B_{16} \\ \hline C5_{16} \end{array}$$

Wenn Sie vergessen haben, wie die Subtraktion funktioniert, dann lesen Sie ab der Seite S 37 noch einmal nach. Auch der Mikroprozessor führt diese und die folgende Rechnung durch (Versuche S 43.1 und S 41.1).

Jetzt wird addiert:

$$\begin{array}{r} C8_{16} \\ + C5_{16} \\ \hline 18D_{16} \end{array}$$

An der 16_{10}^2 -Stelle, also an der (von rechts gezählt) dritten Stelle, wird eins abgezogen und es ergibt sich – wie bei der Subtraktion – $8 D_{16}$.

Zu Aufgabe S 56.1

$$\begin{array}{r}
 \text{a) } \begin{array}{r} 100101000111 \\ - 001001111000 \\ \hline \end{array} \begin{array}{l} \text{Subtrahend} \\ \text{? ? ? ? ? ? ? ? ? ?} \end{array} + \begin{array}{r} 100101000111 \\ + 110110001000 \\ + 11 \\ \hline \end{array} \begin{array}{l} \text{Zweierkomplement} \\ \text{Übertrag} \end{array} \\
 \hline
 \end{array}$$

Die ganz links stehende 1 wird weggestrichen. Das Ergebnis der Subtraktion ist $(0)1101100111_2 = 1743_{10}$

Wir verzichten auf die Angabe der ausführlichen Umrechnung in die Dezimal-Darstellung gemäß dem Schema im Bild S 46.2.

b) Neunzehn = 10011_2 ; elf = 1011_2

Der Subtrahend wird mit vorgestellten Zeichen 0 zur Stellenzahl des Minuenden aufgefüllt: 01011_2 ; Zweierkomplement: 10101_2

$$\begin{array}{r}
 10011 \\
 + 10101 \\
 \hline
 101000
 \end{array}
 \quad (0)1000_2 = 8_{10}$$

c) $2375_{10} = 100101000111_2$; $116_{10} = 1110100_2 = 000001110100_2$
Zweierkomplement: 111110001100

$$\begin{array}{r}
 100101000111 \\
 + 111110001100 \\
 \hline
 1100011010011
 \end{array}
 \quad 100011010011_2 = 2259_{10}$$

d) $F6_{16} = 246_{10} = 11110110_2$; $9C_{16} = 156_{10} = 10011100_2$
Zweierkomplement: 01100100_2

$$\begin{array}{r}
 11110110 \\
 + 01100100 \\
 \hline
 101011010
 \end{array}
 \quad (0)1011010_2 = 90_{10}$$

Sie werden noch sehen, daß diese Teilaufgabe d die einfachste von allen ist.

Zu Aufgabe H 89.1

$$\begin{array}{r}
 \text{a) } \begin{array}{r} 72 \\ + 3C \\ \hline \end{array} \begin{array}{l} \text{A E} \end{array} \quad \text{b) } \begin{array}{r} 0111 \quad 0010 \\ + 0011 \quad 1100 \\ \hline 11 \end{array} \begin{array}{l} \text{Übertrag} \\ 1010 \quad 1110 \end{array}$$

c) $AE_{16} = 174_{10}$; $1010\ 1110_2 = 174_{10}$

Die Addition der dezimal dargestellten und der dual dargestellten Zahlen liefert das gleiche Ergebnis: Einhundertvierundsiebzig

Zu Aufgabe H 91.1

Der Befehl XPAH 1 transportiert das HOB aus P1 in den Akkumulator und den Inhalt AD des Akkumulators als HOB nach P1.

Der Befehl CAS kopiert den Inhalt C5 des Akkumulators in das Status-Register.

Es ergibt sich: (AC) = C5; (P1) = AD 46

Der Befehl XPAL 1 transportiert das LOB aus P1 in den Akkumulator und den Inhalt C5 des Akkumulators als LOB nach P1.

Es ergibt sich: (AC) = 46; (P1) = AD C5.

Das Byte C5 hat also eine Rundreise gemacht und ist im LOW-Teil des Pointer-Registers P1 gelandet.

Der letzte Befehl CAS kopiert den Inhalt 46 des Akkumulators in das Status-Register.

Das Bild Ü 23.1 zeigt die Bitmuster am Ende des Programms.

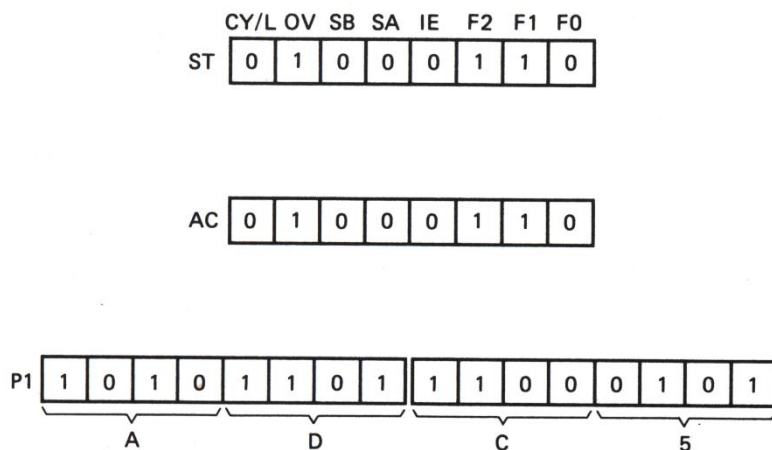


Bild Ü 23.1

Zu Aufgabe H 91.1. Die hier dargestellten Bitmuster ergeben sich nach Ablauf des zum Versuch H 90.1 gehörenden Programms im Status-Register, im Akkumulator und im Pointer-Register 1.

Zu Aufgabe H 94.1

Für das Leuchten bestimmter Lumineszenzdioden ist das vom Akkumulator in das Status-Register kopierte Bitmuster zuständig. Mit den Befehlen ① und ④ werden diese Bitmuster bestimmt.

Wenn zuerst die Lumineszenzdiode F0 leuchten soll, dann muß das Bit F0 den Wert 1 annehmen; alle anderen Bits müssen die Werte 0 haben. Das wird durch das Bitmuster 0000 0001 = 01_{16} erreicht (Bild H 86.1).

Der Akkumulator muß also durch den Befehl ① mit 01 geladen werden: LDI 01. Mit dem Operationscode ergibt sich: C4 01.

Wenn anschließend die Lumineszenzdioden F 2 und F 1 leuchten sollen, dann müssen die Bits F 2 und F 1 die Werte 1 annehmen, alle anderen Bits müssen die Werte 0 haben: $0000\ 0110 = 06_{16}$.

Der Akkumulator muß durch den Befehl ④ mit 06 geladen werden: LDI 0 6. Mit dem Operationscode ergibt sich: C4 06.

Bei beiden Befehlen braucht nur das dem Operationscode folgende Byte geändert zu werden.

Versuch Ü 24.1

Änderung des Blink-Programms vom Versuch H 93.1

TASTEN	ANZEIGE	BEMERKUNGEN
RESET	0420 C4	Anfangsstellung
ME	0421 02	Befehlsnummer für zweites Byte in LDI 02
0, 1, IN	0422 07	LDI 01
4, 2, 8, AD	0428 05	Befehlsnummer für zweites Byte in LDI 05
0, 6, IN	0429 07	LDI 06
RESET	0420 C4	Anfangsstellung
GO		Programm-Start.

Die Lumineszenzdioden leuchten in der gewünschten Folge, beginnend mit F0.

Prüfungsaufgaben Lehrbrief 4

Bitte beachten Sie beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü 1 und Ü 2.

1. Tragen Sie in die Tabelle auf dem Lösungsblatt die Displacements für die geforderten **PC-relativen, unbedingten Sprünge** in die rot unterlegten Felder ein. In die Tabelle sind die Adressen für das Byte 90 des Sprungbefehls eingetragen, sowie die Adressen des nächsten, nach dem Sprung auszuführenden Befehls.

Beachten Sie: Nicht alle geforderten Sprünge sind PC-relativ möglich! – Schreiben Sie in der auf die Tabelle folgenden Aufstellung jeweils an: „Möglich“ oder „Nicht möglich, weil...“.

2. Auf dem Lösungsblatt finden Sie unter a), b) und c) die Ausschnitte aus drei Programm-Tabellen. In allen drei Fällen soll mit einem Sprungbefehl in den Speicherzellen mit den Adressen 0430 und 0431 die Fortsetzung des Programms bei einem „nächsten Befehl“ veranlaßt werden. In allen drei Fällen ist die Adresse dieses nächsten Befehls vom Sprungbefehl soweit entfernt, daß sie nicht PC-relativ erreicht werden kann. Es müssen also **Pointer-indizierte Sprungbefehle** verwendet werden.

In allen drei Fällen ist (P1) = 05E6 und (P2) = 0530.

Wählen Sie jeweils das Pointer-Register aus, das die Adresse beinhaltet, von der aus ein Vorwärts- oder Rückwärtssprung zum nächsten Befehl möglich ist, und tragen Sie jeweils in das rot unterlegte Feld den Operationscode und das Displacement für den Sprung ein (Seite S 77).

3. Sie sollen das Programm für eine Verkehrsampel schreiben.

Zunächst wollen wir uns damit begnügen, diese Verkehrsampel im Zeitraffer-Tempo ablaufen lassen. In einem späteren Lehrbrief werden wir Ihnen einen Trick zeigen, mit dem man ein solches Programm mit einer erträglichen Anzahl von Befehlen auch beliebig langsam ablaufen lassen kann.

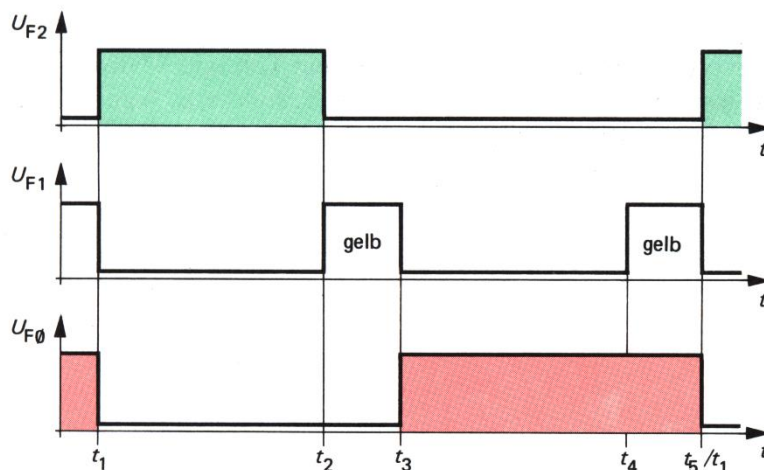


Bild Ü 25.1
Zu Prüfungsaufgabe 3: Zeitlicher Verlauf für das Leuchten der Lampen in einer Verkehrsampel

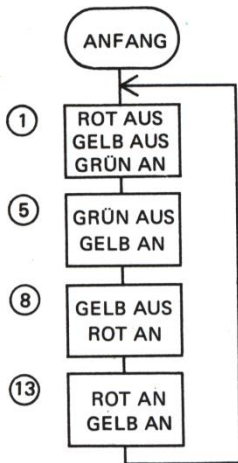


Bild Ü 26.1
Zu Prüfungsaufgabe 3: Programmab-
laufplan einer Verkehrsampel

Im Bild Ü 25.1 haben wir den zeitlichen Verlauf für das Leuchten der einzelnen Lampen eingetragen: Zunächst leuchtet die grüne Lampe vom Zeitpunkt t_1 bis zum Zeitpunkt t_2 .

Zum Zeitpunkt t_2 wird die grüne Lampe abgeschaltet und sofort anschließend die gelbe Lampe angeschaltet. Die gelbe Lampe leuchtet bis zum Zeitpunkt t_3 .

Zum Zeitpunkt t_3 wird die gelbe Lampe abgeschaltet und sofort anschließend die rote Lampe angeschaltet. Kurz vor ihrem Verlöschen (Zeitpunkt t_4) wird die gelbe Lampe zugeschaltet. Zwischen den Zeitpunkten t_4 und t_5 leuchten die gelbe und die rote Lampe gemeinsam.

Der Zeitpunkt t_5 ist wieder mit dem Zeitpunkt t_1 identisch: In diesem Augenblick verlöschen die gelbe und die rote Lampe und die grüne Lampe wird allein angeschaltet. Ab jetzt wiederholt sich der beschriebene Vorgang.

Im Bild Ü 26.1 haben wir den Ablaufplan für das hier beschriebene Programm dargestellt; allerdings ohne Rücksicht auf die Befehle unseres Mikroprozessors.

Benützen Sie im Programm das Signal FLAG 2 zur Steuerung der grünen Lampe, das Signal FLAG 1 zur Steuerung der gelben Lampe und das Signal FLAG 0 zur Steuerung der roten Lampe! – Auf den Seiten H 90 bis H 93 haben wir die Programmierung eines ähnlichen Vorgangs beschrieben!

- Ergänzen Sie den Programmablaufplan auf dem Lösungsblatt unter Berücksichtigung der Befehle des Mikroprozessors! Nehmen Sie ähnliche Eintragungen vor wie im Bild H 93.1! – Die Leuchtdauer der gelben Lampe wird durch einen einmaligen Verzögerungsbefehl DLY FF (8F FF) bestimmt; die rote Lampe allein und die grüne Lampe sollen jeweils dreimal so lange leuchten.
- Tragen Sie das entsprechende Programm in die Liste auf dem Lösungsblatt ein! Tragen Sie in die grün unterlegten Spalten die Befehle in mnemonischer Schreibweise ein und in die rot unterlegten Spalten die Operationscodes und die zweiten Bytes bei Zwei-Byte-Befehlen. Berechnen Sie das Displacement des Rücksprungs zum Anfang des Programms!
- Welche Buchsen müssen Sie auf der Leiterplatte für Relais und LEDs miteinander verbinden, wenn Sie die farbigen Lumineszenzdiode sinnngemäß im Versuch verwenden wollen?

- Warum ergibt sich im Versuch S 90.1 kein Ton oder Halbton der Tonleiter, wenn Sie zwei nicht benachbarte Tasten (z.B. 1 und 3 oder 5 und 7) gleichzeitig betätigen?

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe H 105.1

Das letzte Byte des letzten Befehls (00) ist unter der Nummer $04C7_{16}$ abgelegt; das erste Byte des ersten Befehls steht unter der Nummer 0420_{16} . Die Anzahl der Bytes ergibt sich also aus der Differenz

$$(0)4C7_{16} - (0)41F_{16} = A8_{16} = 168_{10}$$

Das Uhrenprogramm enthält also einhundertachtundsechzig Bytes.

Zu Aufgabe S 60.1

Man kann den Befehl LDE benutzen! (Bild H 63.1 und Seite H 64.) Im Gegensatz zum Austausch-Befehl XAE handelt es sich um einen Kopier-Befehl.

Da es im hier besprochenen Fall nur darauf ankommt, der Inhalt des Extension-Registers in den Akkumulator zu bringen, sind beide Befehle gleichwertig.

Zu Aufgabe S 63.1

Selbstverständlich nehmen Sie den Austausch per Programm vor! Denn für diesen Austausch wurde das Programm ja entworfen!

Zu beachten ist allerdings, daß die Anhalte-Anweisung neu geladen werden muß:

TASTEN	ANZEIGE	BEMERKUNGEN
4, 3, 6, AD	0436 XX	Anwahl der Adresse für die Anweisung „Anhalten“.
BP	0436 3F	Anfangsstellung. Hier können Sie sich das Betätigen des Drucktastschalters RESET auch sparen: Bei GO startet das Programm immer bei 0420.
GO	0420 C4	Austausch.

Zu Aufgabe S 67.1

Der Akkumulator wird mit der sedezimal dargestellten Zahl geladen, die um eins kleiner ist als B0, also mit AF. Anschließend wird der Inhalt des Akkumulators mit dem Inhalt des LOW-Teils des Pointer-Registers PC ausgetauscht. — Es ergibt sich folgende Anweisung:

0420 LDI AF (Zwei-Byte-Befehl)
 0422 XPAL 0 (Ein-Byte-Befehl)

Sie können sich von der Richtigkeit der Überlegung im Versuch überzeugen:

Versuch Ü 28.1

Die Änderung des PC-Inhalt bewirkt einen Programm-Sprung

Den Operations-Code des Befehls LDI haben Sie auf der Seite H 84 kennengelernt: C4. Das Displacement (Seite S 61) enthält das um eins verminderte, neue LOB des Pointer-Registers PC: AF.

Die Operations-Code des Befehls XPAL 0 lautet 30.

TASTEN	ANZEIGE	BEMERKUNGEN
RESET	0420 XX	Anfangsstellung
C, 4, IN		
A, F, IN	0422 XX	LDI AF Akku mit AF laden
3, 0, IN	0423 XX	XPAL 0
RESET	0420 C4	Anfangsstellung
SI	0422 30	Akku wird geladen
AC	0000 AF	(AC) = AF
SI	04B0 XX	Adresse 04B0 für nächsten Befehl

Erinnern Sie sich: An den ersten vier Stellen der Anzeige steht immer der aktuelle Inhalt des Pointer-Registers PC (Befehlszähler).

Zu Aufgabe S 74.1

Für beliebig weite Sprünge kann man eins der Pointer-Register mit der Ziel-Adresse laden, die genau um eins kleiner ist, als die Adresse des nächsten, auszuführenden Befehls. Der Befehl XPPC (Seite H 78) löst dann den beabsichtigten Sprung aus. — Sie werden gleich noch eine andere, elegantere Methode für beliebig weite Sprünge kennenlernen.

Zu Aufgabe S 74.2

	a)	b)	c)	d)	e)	f)	g)
90	043A	0450	0526	051C	04E3	13C4	7F8C
Disp.							
Nächster Befehl	043E	042B	05CD	04E3	0510	1393	8021
Displacement	02	D9	(A5) nicht möglich	C5	2B	CD	nicht möglich

Zu Aufgabe S 75.1

Wenn die Lumineszenzdioden F2 und F0 aufleuchten sollen, muß im Status-Register das im Bild Ü 29.1 dargestellte Bitmuster stehen. Dieses Byte wird dezimal als 05 dargestellt.

Per Programm wird also der Akkumulator mit 05 geladen und dieser Inhalt ins Status-Register kopiert.

Damit die Bitmuster-Anzeige des Inhalts des Status-Registers sichtbar wird, folgt eine Verzögerung (Seite H 92).

Das anschließende Laden des Akkumulators mit 00 und das Kopieren dieses Bytes ins Status-Register setzt alle Bits des Status-Registers auf 0; die Lumineszenzdioden verlöschen.

Ein weiterer Verzögerungsbefehl macht das Verlöschen sichtbar. Damit anschließend die Lumineszenzdioden wieder aufleuchten, erfolgt ein Rückwärtssprung zum neuerlichen Durchlaufen des Programms.

Das Bild Ü 29.2 zeigt das aufgelistete Programm.

Die eingetragenen Adressen erlauben das Berechnen des Displacements. Die Start-Adresse für den Rückwärts-Sprung ist die des zweiten Bytes des Sprungbefehls: 050B.

Die Ziel-Adresse ist um eins kleiner als die Adresse 0500 des Programm-Anfangs: 04FF.

Ziel-Adresse: 4 F F

Start-Adresse: — 5 0 B

Displacement: (F)F 4

Bei unserem Mikroprozessor bieten die Anschlüsse FLAG0 und FLAG2 die einfachste Möglichkeit, Signale an die Peripherie zu liefern. Wir werden in späteren Versuchen noch häufiger Programme des gleichen Prinzips verwenden, das Sie hier selbständig gehandhabt haben.

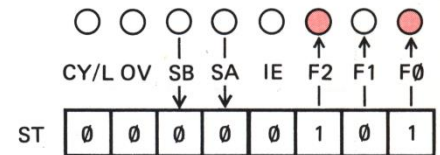


Bild Ü 29.1

Zu Aufgabe S 75.1: Wenn die Lumineszenzdioden F0 und F2 leuchten sollen, dann muß das Status-Register das hier dargestellte Bitmuster enthalten.

Nr.	Befehl		Adresse (PC)	Opc.	Disp.	Erläuterungen
1	LDI		0500	C4		F0, F2 an
		05	0501		05	
2	CAS		0502	07		Verzögern
3	DLY		0503	8F		
		FF	0504		FF	
4	LDI		0505	C4		F0, F2 aus
		00	0506		00	
5	CAS		0507	07		Verzögern
6	DLY		0508	8F		
		FF	0509		FF	
7	JMP		050A	90		Sprung z. Anfang bei 0500
		0500	050B		F4	

Bild Ü 29.2

Zu Aufgabe S 75.1: Blink-Programm für die Lumineszenzdioden F0 und F2.

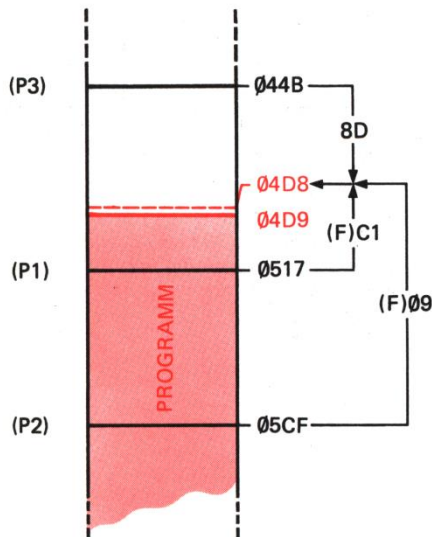


Bild Ü 30.1

Zu Aufgabe S 79.1: Lage der in den Pointer-Registern P1, P2 und P3 gespeicherten Adressen und der Adresse für den Beginn des fortzusetzenden Programms. Wir haben die jeweiligen Sprungweiten eingetragen.

Zu Aufgabe S 76.1

Ziel-Adresse: 4 F F
 Start-Adresse: $\underline{-4\ 2\ 1}$
 Displacement: (0) D E

An der (von rechts gezählt) dritten Stelle steht der Wert 0; zunächst scheint der Vorwärts-Sprung möglich.

An der (von rechts gezählt) zweiten Stelle steht ein Wert zwischen 8 und F. Da ein Vorwärtssprung ausgeführt werden soll, ist der Sprung PC-relativ nicht möglich.

Zu Aufgabe S 79.1

- a) Es muß festgestellt werden, ob von einer der in den Pointer-Registern gespeicherten Adressen ein Sprung zur Ziel-Adresse $04D8_{16} = 04D9_{16} - 1_{16}$ möglich ist. Die Berechnung der Sprungweiten ergibt:

	P3	P1	P2
Ziel-Adresse:	4 D 8	4 D 8	4 D 8
Start-Adresse:	$\underline{-4\ 4\ B}$	$\underline{-5\ 1\ 7}$	$\underline{-5\ C\ F}$
Sprungweite:	(0) 8 D	(F) C 1	(F) 0 9

Der Vorwärtssprung von der in P3 gespeicherten Adresse ist nicht möglich, weil sich bei der Sprungweite an der (von rechts gezählt) zweiten Stelle ein größerer Wert als 7 ergibt.

Der Rückwärtssprung von der in P1 gespeicherten Adresse ist möglich, weil sich bei der Sprungweite an der (von rechts gezählt) zweiten Stelle ein Wert zwischen 8 und F ergibt.

Der Rückwärtssprung von der in P2 gespeicherten Adresse ist nicht möglich, weil sich bei der Sprungweite an der (von rechts gezählt) zweiten Stelle ein kleinerer Wert als 8 ergibt.

Im Bild Ü 30.1 haben wir in die Darstellung des Speichers rot das fortzusetzende Programm mit seiner Anfangsadresse 04D9 und die in den Pointer-Registern P1, P2 und P3 gespeicherten Adressen eingezeichnet. Zusätzlich finden Sie die jeweiligen Sprungweiten eingetragen.

Der Inhalt des Pointer-Registers P1 ist ohne Änderung als Sprungbrett für einen Pointer-indizierten Sprung verwendbar.

- b) Der Operationscode für einen Pointer-indizierten Sprung mit der Start-Adresse in P1 ist 91. Das Displacement ergibt sich unmittelbar aus der Sprungweite. Der Sprungbefehl lautet also: 91 C1.

Prüfungsaufgaben Lehrbrief 5

Beachten Sie bitte beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü 1 und Ü 2.

1. Nehmen Sie an, das im Bild S 96.1 aufgelistete Blinkprogramm „NEG“ würde nicht bei der Adresse 0480 beginnen, sondern der erste Befehl dieses Blinkprogramms (CSA) stünde an der Adresse 0456, gleich anschließend an den Sprungbefehl am Ende des Rechenprogramms.

Ergänzen Sie für diesen Fall den auf dem Lösungsblatt unvollständig angegebenen Programmablaufplan.

2. Die Lösung der Aufgabe S 102.1c ergab, daß für sechsmaliges Aufleuchten der Lumineszenzdiode F2 bei positivem Rechenergebnis bei der Adresse 0461 im Bild S 101.1 der Wert $0C_{16} = 12_{10}$ eingesetzt werden muß.

$0C_{16} = 12_{10}$ ist eine gerade Zahl; sie ist durch zwei ohne Rest teilbar.

- a) Was ist die Folge, wenn Sie bei der Adresse 0461 eine dezimal dargestellte ungerade Zahl, z.B. $0B_{16} = 11_{10}$, eingeben?

Machen Sie einen Versuch! (Programm-Liste im Bild S 103.1.) Bilden Sie nacheinander mehrere Differenzen mit positiven Ergebnissen und beobachten Sie die Lumineszenzdiode F2!

- b) Geben Sie den Grund für das Verhalten der Lumineszenzdiode F2 an!

3. Beim Hauptprogramm (Bild S 114.1) und beim Unterprogramm NEG (Bild S 114.2) für den Versuch S 114.1 haben wir die Sprungbefehle entsprechend den Programmablaufplänen in den Bildern S 113.1 (Hauptprogramm) und S 115.1a (Unterprogramm) gewählt.

Für den Sprung vom Hauptprogramm zum Unterprogramm kann wahlweise auch entsprechend dem Bild S 112.1 programmiert werden, und für den Sprung vom Unterprogramm zum Hauptprogramm kann die im Bild S 115.1b dargestellte Version gewählt werden.

Auf dem Lösungsblatt haben wir das Programm für den Versuch S 114.1 abgedruckt (vgl. Bild S 114.2). Die Sprungbefehle am Ende des Hauptprogramms und des Unterprogramms haben wir nicht eingetragen.

Ergänzen Sie das Programm so, daß die fehlenden Sprungbefehle am Ende des Hauptprogramms (042F bis 0432) dem Programmablaufplan im Bild S 112.1 entsprechen.

Die Sprungbefehle am Ende des Unterprogramms sollen dem Programmablaufplan im Bild S 115.1b entsprechen.

4. Ändern Sie das Programm für den Tongenerator im Bild S 119.1 so, daß der für den Versuch S 123.1 angeschlossene Lautsprecher einen Ton mit der Frequenz $f = 850 \text{ Hz}$ abgibt!

Geben Sie auf dem Prüfungsblatt nur die Adresse und das Byte an, die für die gewünschte Frequenz verantwortlich sind!

5. Bei einer Verkehrsampel leuchtet die gelbe Lampe jeweils — unabhängig von der Dauer der Rot- und Grün-Phase — im praktischen Betrieb etwa 3 s. (Genau genommen ist die Gelb-Phase noch etwas kürzer.)

Ändern Sie das Programm für die Verkehrsampel auf der Seite S 127 so, daß beim Versuch S 128.1 die gelbe Lumineszenzdiode jeweils für 3 s aufleuchtet.

Gehen Sie davon aus, daß in unserem Programm die Leuchtdauer der Lumineszenzdioden praktisch allein von der Verzögerungsschleife bestimmt wird (Seite S 129). Nehmen Sie für Ihre Überlegungen das Bild S 128.1 zu Hilfe!

Geben Sie als Lösung nur die Adressen der Speicherzellen im Programm auf der Seite S 127 an, bei denen eine Änderung notwendig ist, und schreiben Sie direkt neben die jeweilige Adresse das neue Byte an.

6. Beim Versuch S 139.1 können Sie die Frequenzen der generierten Töne sämtlich um eine Oktave ändern, wenn Sie auf der Leiterplatte für Relais und LEDs die von der Buchse A1 kommende Leitung von FLAG0 nach FLAG1 umstecken.

Geben Sie den Grund für diese Erscheinung an!

(Lesen Sie die Anweisungen zum Versuch S 84.1 noch einmal durch!)

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe S 97.1

a) Natürlich hat der Mikroprozessor richtig gerechnet:

(1)	0	1	2	16
	F	10	12	
—	0	A	B	16
	F	6	7	16

Das Zeichen F an der (von rechts gezählt) dritten Stelle weist das Ergebnis als negativ aus (vgl. ab Seite S 72). Das Zeichen wird bei zweistelliger Darstellung der Zahlen nicht angezeigt. An den (von rechts gezählt) zweiten und ersten Stellen erscheinen die Ziffern 67_{16} . Dieses Byte, zu dem das Bitmuster 01100111 gehört, gehört zu einer negativen Zahl, weist aber bei zweistelliger Darstellungsweise diese Zugehörigkeit nicht aus: Sein MSB hat den Wert 0. In geradezu betrügerischer Absicht nutzt es aus, daß man das zugehörige Zeichen F an der (von rechts gezählt) dritten Stelle nicht sehen kann. Es benimmt sich wie ein Hochstapler und gaukelt uns vor, es stelle eine positive Zahl dar. Der „intelligente“ Mikroprozessor fällt darauf herein und springt ins POS-Programm.

b) Die Grenzen der Fähigkeiten unseres Programms werden überschritten, wenn im Bitmuster des sedezimal zweistellig dargestellten Rechenergebnisses bei negativen Zahlen das MSB den Wert 0 annimmt. Das ist bei der Zahl „minus Einhundertneundzwanzig“ der Fall. Diese Zahl müßte entsprechend der Tabelle im Bild S 73.2 als $7F_{16}$ dargestellt werden. Das ist aber die sedezimale Darstellung der Zahl „plus Einhundertsiebenundzwanzig“.

Eine entsprechende Überlegung führt übrigens zu dem Ergebnis, daß die Grenzen der Fähigkeiten unseres Programms dann überschritten werden, wenn im Bitmuster des sedezimal zweistellig dargestellten Rechenergebnisses bei positiven Zahlen das MSB den Wert 1 annimmt. Das ist bei der Zahl „plus Einhundertachtundzwanzig“ der Fall. (Vgl. Bild S 73.2.)

Lassen Sie Ihr System das Ergebnis der Rechnung $FA_{16} - 6E_{16}$ anzeigen! (RESET, GO, Minuend, IN, Subtrahend, ME.) Bei dieser Rechnung ist der Minuend größer als der Subtrahend: $FA_{16} - 6E_{16} = 8C_{16}$. Das Ergebnis ist positiv. Trotzdem springt unser Programm wegen des Werts 1 des MSB im Byte 8C zum Programmteil NEG; die Lumineszenzdiode F0 täuscht uns ein negatives Ergebnis vor.

Das Programm kann bei der zweistelligen Zahlendarstellung nur solche Rechnungen richtig ausführen, bei denen das Ergebnis zwischen $80_{16} = -128_{10}$ und $7F_{16} = +127_{10}$ liegt.

Zu Aufgabe S 100.1

Würde man das Setzen des Zählers für die Schleifendurchläufe in die Schleife einbeziehen, dann könnte man sich den ganzen Aufwand mit dem Zähler auch sparen. Der Zähler würde zwar jeweils von der Anweisung DLD dekrementiert; er käme aber niemals bei null an, weil er ja bei Beginn des Schleifendurchlaufs jedesmal auf seinen ursprünglichen Wert zurückgesetzt würde!

Zu Aufgabe S 102.1

- a) Das Displacement ergibt sich zu $42D_{16} - 46F_{16} = BE_{16}$
 b) Das Displacement ergibt sich zu $463_{16} - 471_{16} = F2_{16}$
 c) Die Lumineszenzdiode wird beim ersten Schleifendurchlauf eingeschaltet, beim zweiten Schleifendurchlauf ausgeschaltet, beim dritten wieder eingeschaltet usw. (Vgl. ab Seite S 83).
 Für einmaliges Aufleuchten und Verlöschen sind also zwei Schleifendurchläufe notwendig, für sechsmaliges Aufleuchten deren zwölf.
 Der Zähler muß demnach durch das Displacement bei der Adresse 0461 auf $12_{10} = 0C_{16}$ gesetzt werden.

Bild Ü 34.1

Zu Aufgabe S 103.1: Liste des Programmteils NEG zum Voreinstellen der Blinkhäufigkeit der Lumineszenzdiode FO.

Nr.	Label	Befehl		Adresse (PC)	Opc.	Disp.	Erläuterungen
1	NEG	LDI		0480	C4		Zähler setzen
			ANZ	0481		0C	
2		ST		0482	C8		
			ZÄHL	0483		0F	
3	BLIN	CSA		0484	06		FO blinkt
4		ADI		0485	F4		
			01	0486		01	
5		ANI		0487	D4		
			01	0488		01	
6		CAS		0489	07		
7		DLY		048A	8F		
			FF	048B		FF	
8		DLD		048C	B8		Zähler dekrementieren und laden
			ZÄHL	048D		05	
9		JZ		048E	98		(Zähler) = 0?
			RECH	048F		9E	Ja: Sprung → Rechenprogramm
10		JMP		0490	90		Nein: Sprung nach BLIN, neuer Schleifendurchlauf
			BLIN	0491		F2	
11	ZÄHL			0492			Zähler (variable Daten)

Zu Aufgabe S 103.1

Bild Ü 34.1

Damit die Lumineszenzdiode F0 leuchtet, muß durch den Befehl ADI jeweils zum Bit Nr. 0 (LSB) der Wert 1 addiert werden (ADI 01). Der anschließende Befehl ANI hat ebenfalls das Displacement 01 (vgl. Seite S 95.)

Das Displacement des Befehls für den unbedingten Rücksprung zum Rechenprogramm ergibt sich zu $42D_{16} - 48F_{16} = 9E_{16}$.

Im übrigen sind die Programmteile NEG und POS gleich.

Die Speicherzelle mit der Adresse 0492 enthält keinen Befehl, also keine Programm-Daten, sondern die Zahl, die dem jeweiligen Zählerstand entspricht. (Vgl. Seite H 106.)

Zu Aufgabe S 105.1

Im Bild Ü 35.1 finden Sie die Befehlsliste für das Programm.

Das Pointer-Register P 3 wird mit der um eins verminderten Anfangsadresse 0480 des Unterprogramms NEG geladen, also mit 047F. – Die erste Adresse der Lade-Anweisung für P3 haben wir mit dem Label ANFA gekennzeichnet.

Die Warteschleife (Label WARTE) besteht aus den Befehlen CSA, ANI 30 (Aussortieren von SA und SB) und dem Befehl JZ für den bedingten Sprung bei (AC) = 00 zum Unterprogramm.

Der Befehl XPPC 3 bringt die um eins verminderte Anfangsadresse des Unterprogramms aus dem Pointer-Register P3 in dem Befehlszähler PC.

Bild Ü 35.1

Zu Aufgabe S 105.1: Programm zum Laden des Pointer-Registers P3, für eine Warteschleife und für den Sprung zum Unterprogramm.

Nr.	Label	Befehl		Adresse (PC)	Opc.	Disp.	Erläuterungen
1	ANFA	LDI		0420	C4		Adresse Unterprogramm nach P3
			04	0421		04	
2		XPAH	(P3)	0422	37		
3		LDI		0423	C4		
			7F	0424		7F	
4		XPAL	(P3)	0425	33		
5	WARTE	CSA		0426	06		Warteschleife
6		ANI		0427	D4		
			30	0428		30	
7		JZ		0429	98		
			WARTE	042A		FB	
8		XPPC 3		042B	3F		Sprung zum Unterprogramm

Zu Aufgabe S 105.2

- a) Beim Befehl 9. JZ RECH muß der Inhalt der Speicherzelle mit der Adresse 048F so geändert werden, daß sich der Befehl JZ ANFA ergibt.
- b) Der Sprung muß zu der um eins verminderten Anfangsadresse 0420 erfolgen: $041F_{16} - 048F_{16} = (F) 90_{16}$
- c) Auch beim Programmteil POS muß der Befehl 9. JZ RECH in den Befehl JZ ANFA geändert werden. Bei diesem Programmteil steht das entsprechende Displacement in der Speicherzelle mit der Adresse 046F. Für ihren neuen Inhalt ergibt sich: $041F_{16} - 046F_{16} = (F) B0_{16}$

Zu Aufgabe S 107.1

Statt des Programmteils NEG muß jetzt der Programmteil POS angesprungen werden. Das Pointer-Register P3 muß also statt mit der Adresse 047F jetzt mit dem Befehl Nr. 3 (Bild S 106.1) mit der Adresse 045F geladen werden. Dazu brauchen Sie nur bei der Adresse 0424 das Byte 7F mit dem Byte 5F zu überschreiben:

RESET, 4, 2, 4, AD, 5, F, IN, (GO).

Zu Aufgabe S 110.1

Im Programmablaufplan bedeutet bei einer Verzweigung „JA“, daß die Sprungbedingung erfüllt ist, daß der entsprechende Sprung also ausgeführt wird. Bei „NEIN“ ist die Sprungbedingung nicht erfüllt; der Sprungbefehl wird übergangen und der nächstfolgende Befehl ausgeführt.

Das Bild Ü 36.1 zeigt den Ablaufplan des Unterprogramms bei Verwendung des Befehls JZ. Solange der Inhalt des Zählers nicht null ist, wird der Befehl für den bedingten Sprung übergangen. Es muß ein Befehl für einen unbedingten Sprung zum Label BLIN folgen.

Wenn der Inhalt des Zählers gleich Null ist, dann wird der bedingte Sprung zum Befehl XPPC 3 ausgeführt.

Wenn wir die sich aus den Programmablaufplänen in den Bildern Ü 36.1 und S 108.1 ergebenden Befehlsfolgen ab dem Befehl DLD ZÄHL nebeneinander stellen, dann ergibt sich folgender Vergleich:

Mit Befehl JZ

DLD	ZÄHL
JZ	XPPC
JMP	BLIN
XPPC	3
JMP	NEG

Mit Befehl JNZ

DLD	ZÄHL
JNZ	BLIN
XPPC	3
JMP	NEG

Bei Verwendung des Befehls JNZ läßt sich also ein Zwei-Byte-Befehl einsparen.

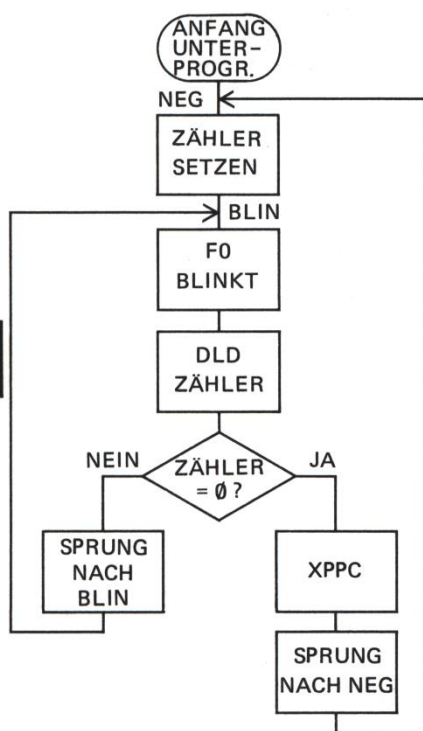


Bild Ü 36.1
Zu Aufgabe S 110.1: Bei Verwendung des Befehls JZ wird ein Zwei-Byte-Befehl mehr gebraucht als bei Verwendung des Befehls JNZ. (Vgl. Bild S 108.1.)

Wenn Sie die Befehlsfolge mit dem Befehl JZ ausprobieren wollen, dann müssen Sie dem Zähler die Speicherzelle mit der Adresse 0495 zuweisen. Es ergibt sich in der Programmliste im Bild S 109.1 für das Displacement des Befehls ST bei der Adresse 0483 der Wert 12 und für das Displacement des Befehls DLD bei der Adresse 048D der Wert 08.

Zu Aufgabe S 114.1

- a) Im Pointer-Register P1 wird die Adresse 0400 gespeichert und im Pointer-Register P2 die Adresse 05FF.
- b) Weil die in den Pointer-Registern gespeicherten Adressen als Startadressen für Pointer-indizierte Sprünge benutzt werden, müssen die Displacements des Befehls JNZ NEG (P2) bei der Adresse 0430 im Hauptprogramm und JZ WARTE (P1) bei der Adresse 05FA im Unterprogramm geändert werden.

Für das Displacement des Befehls JNZ NEG (P2) ergibt sich:

$$5 \text{ EA}_{16} - 5\text{FF}_{16} = \text{EB}_{16}$$

und für das Displacement des Befehls JZ WARTE (P1) berechnen wir:

$$42\text{B}_{16} - 400_{16} = 2\text{B}_{16}$$

Es ergeben sich also folgende Änderungen:

0430 EB; 05FA 2B

Sie können sich im Versuch von der Richtigkeit der Überlegung überzeugen.

Zu Aufgabe H 120.1

Der Speicher faßt $16384_{10} : 1024_{10} = 16$ KBit. Da ein Byte acht Bit enthält, brauchen Sie nur zu rechnen: $16 \text{ KBit} : 8 = 2 \text{ KByte}$

Zu Aufgabe S 120.1

Im Bild Ü 37.1 haben wir die zur Tonschleife im Versuch S 90.1 (Bilder S 87.1 und S 89.1) gehörenden Befehle aufgelistet und die zu diesen Befehlen gehörenden Mikrozyklen angeschrieben. Die Summe der Mikrozyklen für einen Schleifendurchlauf beträgt

$$M_z = 5 + 11 + 6 + 18 + 151 + 18 + 7 + 6 + 9 + 11 = 242$$

Mit $1 M_z = 1,907349 \mu\text{s}$ ergibt sich für eine Halbperiode

$$\frac{T}{2} = 242 \cdot 1,907349 \mu\text{s} \approx 462 \mu\text{s}$$

Daraus erhält man für die Periodendauer:

$$T = 2 \cdot 462 \mu\text{s} = 924 \mu\text{s} = 924 \cdot 10^{-6} \text{ s}$$

Jetzt können wir die Frequenz berechnen:

$$f = \frac{1}{T} = \frac{1}{924 \cdot 10^{-6} \text{ s}} \approx 1082 \text{ Hz}$$

Nr.	Befehl	Mikrozyklen
9	CSA	5
10	ADI	11
11	CAS	6
12	LD	18
13	DLY	151
14	LD	18
15	XAE	7
16	LDE	6
17	JZ	9
18	JMP	11

Bild Ü 37.1

Zu Aufgabe S 120.1: Auflistung der Mikrozyklen, die zu den Befehlen in der Tonschleife im Bild S 89.1 gehören.

Zu Aufgabe S 121.1

- a) Durch den zur Anweisung ① gehörenden Befehl LDI 02 wird $(AC) = 02_{16} = 2_{10}$. Der nachfolgende Befehl CAS in der Anweisung ② ist ein Kopier-Befehl, der den Inhalt des Akkumulators nicht ändert (Seite H 67).

Mit dem Wert $FF_{16} = 255_{10}$ des Displacements des ersten der Verzögerungsbefehle in der Anweisung ③ ergibt sich

$$\frac{t_{v1}}{M_z} = 2 \cdot 2 + 514 \cdot 255 + 13 = 131087$$

Der erste der Verzögerungsbefehle hinterläßt $(AC) = FF_{16} = 255_{10}$. Mit diesem Inhalt des Akkumulators wird der zweite Verzögerungsbefehl mit dem Displacement $FF_{16} = 255_{10}$ ausgeführt:

$$\frac{t_{v2}}{M_z} = 2 \cdot 255 + 514 \cdot 255 + 13 = 131593$$

Die Anweisung ③ bewirkt also eine Verzögerung von insgesamt

$$t_v = t_{v1} + t_{v2} = 131087 M_z + 131593 M_z = 262680 M_z$$

Mit $M_z = 1,907349 \mu s = 1,907349 \cdot 10^{-6} s$ erhalten wir:

$$t_v = 262680 \cdot 1,907349 \cdot 10^{-6} s \approx 0,5 s$$

Die Lumineszenzdiode F1 leuchtet also etwa eine halbe Sekunde lang.

- b) Durch den zur Anweisung ④ gehörenden Befehl LDI 05 wird $(AC) = 05_{16} = 5_{10}$

Dieser Akkumulator-Inhalt ist auch unmittelbar vor dem Verzögerungsbefehl in der Anweisung ⑥ unverändert erhalten. Mit dem Displacement $FF_{16} = 255_{10}$ des DLY-Befehls ergibt sich demnach eine Verzögerung von

$$\frac{t_v}{M_z} = 2 \cdot 5 + 514 \cdot 255 + 13 = 131093$$

Setzen wir $M_z = 1,907349 \cdot 10^{-6} s$ in diese Gleichung ein, dann erhalten wir

$$t_v = 131093 \cdot 1,907349 \cdot 10^{-6} s \approx 0,25 s$$

Die Lumineszenzdioden leuchten also etwa eine viertel Sekunde lang.

Zu Aufgabe S 124.1

1. Für die Periodendauer des 1000-Hz-Tons gilt:

$$T = \frac{1}{f} = \frac{1}{1000 \text{ Hz}} = 10^{-3} s$$

2. Die Schleife muß einmal je Halbperiode durchlaufen werden.

$$\frac{T}{2} = 0,5 \cdot 10^{-3} s = 500 \mu s$$

$500 \mu s$ entsprechen $\frac{500 \mu s}{1,907349 \mu s} \approx 262$ Mikrozyklen.

3. Die Anweisungen in der Schleife (Bild S 119.2) laufen ohne eingebaute Verzögerung in 43 Mikrozyklen ab.
4. Für die Verzögerung ergeben sich $(262 - 43) = 219$ Mikrozyklen.
5. Für das Displacement des DLY-Befehls gilt:

$$\text{Disp}_{10} = \frac{219 - 13}{514} \approx 0,40$$

Es wird nur der ganzzahlige Anteil verwertet:

$$\text{Disp}_{10} = 0; \quad \text{Disp}_{16} = 00$$

Der Akkumulator muß unmittelbar vor dem DLY-Befehl folgenden Inhalt haben:

$$(\text{AC})_{10} = \frac{219 - 13}{2} - 257 \cdot 0 = 103; \quad (\text{AC})_{16} = 67$$

Zur Generierung einer Frequenz von $f = 1000 \text{ Hz}$ muß also die Anweisung (3) in dem im Bild S 119.2 dargestellten Programmablaufplan den Befehl LDI 67 (= C4 67) beinhalten.

Der nachfolgende Verzögerungsbefehl in der Anweisung (4) hat die Form DLY 00 (= 8F 00).

Zu Aufgabe S 126.1

Im Zähler läßt sich als größter Wert $\text{FF}_{16} = 255_{10}$ speichern. — Je Unterprogramm-Aufruf lassen sich also maximal 255_{10} Verzögerungseinheiten aufrufen.

Zu Aufgabe S 128.1

Das erste Setzen des Zählers nach dem Start entsprechend der Anweisung (7) (Bild S 126.1) würde ordnungsgemäß funktionieren, wenn der Speicher-(ST-) Befehl Pointer-3-indiziert programmiert würde.

Nach dem ersten Rücksprung aus dem Unterprogramm (Anweisung (31)) zur Anweisung (10) im Hauptprogramm ist im Pointer-Register P3 nicht mehr die Adresse 0448 abgelegt, sondern die Endadresse 0451 des Unterprogramms (vgl. Seite S 66). Nach dem ersten Abarbeiten des Unterprogramms ändert sich also der Inhalt des Pointer-Registers P3.

Wegen dieser Änderung des Inhalts des Pointer-Registers P3 im Laufe des Programms können bei der gewählten Programmierart keine Pointer-3-indizierten Befehle programmiert werden.

Zu Aufgabe S 138.1

Für eine Verzögerung von 46 Mikrozyklen rechnen wir entsprechend den Anweisungen auf der Seite S 122 zunächst das Displacement des Verzögerungsbefehls DLY aus:

$$\text{Disp}_{10} = \frac{\frac{t_v}{M_z} - 13}{514} = \frac{46 - 13}{514} \approx 0,06$$

Uns interessiert nur der ganzzahlige Anteil: $\text{Disp} = 0_{10} = 00_{16}$

Jetzt berechnen wir den Inhalt des Akkumulators unmittelbar vor dem Verzögerungsbefehl:

$$(\text{AC})_{10} = \frac{\frac{t_v}{M_z} - 13}{2} - 257 \cdot \text{Disp}_{10} = \frac{46 - 13}{2} - 257 \cdot 0 = 16,5$$

Wir runden der Einfachheit halber auf 16 ab und erhalten:

$$(\text{AC}) = 16_{10} = 10_{16}$$

Dieser Wert muß vor dem DLY-Befehl in den Akkumulator geladen werden. Für die Anweisung (24) ergibt sich die Befehlsfolge:

LDI 10, DLY 00.

Prüfungsaufgaben Lehrbrief 6

Beachten Sie bitte beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü1 und Ü2.

1. Bei den Vorüberlegungen zum Versuch S 151.1 (Seite S 150) haben wir das im Bild S 151.1 aufgelistete Programm so konzipiert, daß entsprechend einer der betätigten Tasten 0 bis 7 einer der im Bild S 150.2 zusammengestellten Ziffern-Codes in den Akkumulator geladen wird. Wir benutzen dazu den Pointer-2-indizierten Ladebefehl mit dem Displacement 80. Bei diesem Displacement wird der Wert 80 durch den Inhalt (EX) des Extension-Registers ersetzt (Seite S 149).

Bei welchen Adressen müssen die Ziffern-Codes für den Versuch S 151.1 gespeichert werden, wenn in der Programmliste im Bild S 151.1 der Befehl mit der Nummer 7 wie folgt abgeändert wird:

LDI 80 0429 C4 80 ?

Ergänzen Sie die Adressen in der Tabelle auf dem Lösungsblatt!

Lesen Sie die Aufgabe S 151.1c und deren Lösung auf der Seite Ü 45 noch einmal durch!

2. In dem zum Versuch S 151.1 gehörenden Programm (Bild S 151.1) werden die Tasten in der beim Label BEGIN beginnenden Schleife abgefragt.

Sie sollen herausfinden, wie oft je Sekunde diese Abfrage stattfindet, wie groß also die Abfrage-Frequenz ist.

Lesen Sie den Abschnitt Mikrozyklen als Maß für die Befehlsausführungszeit ab Seite S 117 noch einmal durch!

- a) Wieviele Mikrozyklen braucht unser Mikroprozessor für den einmaligen, vollständigen Durchlauf der Schleife? (Seiten T5 bis T8.)
- b) Wieviele Mikrosekunden braucht unser Mikroprozessor für den einmaligen, vollständigen Durchlauf der Programmschleife?
- c) Wie groß ist die Abfrage-Frequenz, wie oft wird also die Schleife je Sekunde durchlaufen?

3. Im Bild S 156.2 haben wir für den Versuch S 156.1 den autoindizierten Ladebefehl Pointer-1-indiziert programmiert.

Tragen Sie auf dem Lösungsblatt die Befehlsfolge für diesen Versuch mit einem autoindizierten, Pointer-3-indizierten Ladebefehl ein!

4. Das Programm zum Verschieben eines Datenblocks um eine Speicherzelle haben wir mit Rücksicht auf das vom Drucker benötigte Scratchpad (Bild H 115.1) bei der Adresse 05DE enden lassen. Die Adresse des ersten Befehls ist dann 05A9 (Seite S 163).

Wenn man beim Betrieb des Verschiebe-Programms auf die Benutzung des Druckers verzichtet, dann darf dieses Programm bei der Adresse 05FF enden und braucht erst bei der Adresse 05CA zu beginnen.

Geben Sie an, was in der Liste auf der Seite T9 geändert werden muß, wenn das Verschiebe-Programm zwischen den Adressen 05CA und 05FF gespeichert werden soll.

Die Antwort: „Die Adressen müssen geändert werden“ brauchen Sie nicht anzuschreiben.

5. Ergänzen Sie die Tabelle auf dem Lösungsblatt für den Fall, daß die Sieben-Segment-Anzeige die Ziffer 3 mit Dezimalpunkt (3.) anzeigen soll.
6. Begründen Sie bitte, warum in dem zum Versuch H 179.1 entwickelten Programm die Taste GO und nicht die Taste PT zum Auslösen des Druckvorgangs verwendet wird.

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe H 134.1

Im Bild Ü 43.1 haben wir das Programm aus dem Bild H 133.1 in der gewohnten Form aufgelistet.

Mit den ersten vier Befehlen wird das Pointer-Register P3 mit 04FF geladen. — Der letzte Befehl bewirkt den Austausch der Inhalte des Programmzählers und des Pointer-Registers P3, so daß im Programmzähler die Adresse 04FF steht.

Vor der Ausführung des nächsten Befehls wird der Inhalt des Programmzählers um 1 inkrementiert (Seite S 99); als nächster wird also der Befehl ausgeführt, der bei der Adresse 0500 steht.

Operation	Operand	Adresse	Op-code	Displacement	Bemerkungen
LDI	04	0420	C4	04	(Seiten H 75/ H 76)
XPAH 3		22	37		
LDI	FF	23	C4	FF	
XPAL 3		25	33		(Seiten H 75/ H 76)
XPPC 3		26	3F		(Seite H 78)

Bild Ü 43.1
Zu Aufgabe H 134.1

Zu Aufgabe H 153.1

Der Drucker hat bei diesem Versuch keineswegs wirkliche Register-Inhalte der CPU ausgedruckt! Sie haben vielmehr durch direktes Laden der Scratchpad-Zellen 0402 bis 040A einen Teil der Einzelschritt-Anweisung imitiert.

Den Beweis liefert das Status-Register. Beim Anschluß des Druckers leuchtet die Lumineszenzdiode SA auf. Der tatsächliche Inhalt des Status-Registers in der CPU ist also sicher nicht 00, wie es uns der Ausdruck vortun will.

Umgekehrt gelingt es natürlich auch nicht, durch Laden der Scratchpad-Zellen die wirklichen CPU-Register auf die gewünschten Werte zu setzen.

Zu Aufgabe H 154.1

Zuerst wird die Startadresse 0426 für das Laden der Pointer-Register P2 und P3 mit nachfolgendem AD eingetastet.

Die Lade-Anweisung enthält acht Befehle. Die Anzahl der Befehle wird also mit 08₁₆ eingetastet.

Sofort anschließend wird mit 01 die Druckart angegeben und danach der Druckbefehl getastet.

Es ergibt sich also folgende Tastenfolge:

4, 2, 6, AD, 0, 8, 0, 1, PT

Zu Aufgabe H 156.1

Zuerst wird die Adresse 021B des ersten, zu druckenden Bytes mit nachfolgendem AD eingetastet.

Die Anzahl der Bytes ergibt sich aus der um 1 vermehrten Differenz der Adresse des letzten zu druckenden Bytes und der Adresse des ersten zu druckenden Bytes:

$$(02A5_{16} - 021B_{16}) + 1_{16} = 8A_{16} + 1_{16} = 8B_{16}$$

Die Anzahl der Bytes wird also mit 8B eingetastet.

Sofort anschließend wird mit 02 die Druckart angegeben und danach der Druckbefehl getastet.

Es ergibt sich also folgende Tastenfolge:

2, 1, B, AD, 8, B, 0, 2, PT

Zu Aufgabe H 158.1

- a) Der ASCII-String für die erste Zeile endet mit dem Byte 0A (LF, Zeilenvorschub, vgl. Seite T4) bei der Adresse 0518 ab. Die Anfangsadresse für die zweite Zeile ist also 0519.

Der ASCII-String für die zweite Zeile schließt mit dem nächsten Byte 0A für den Zeilenvorschub. Dieses Byte steht bei der Adresse 052C. Da die zweite Zeile ohne nachfolgenden Zeilenvorschub gedruckt werden kann, hört der ASCII-String für die zweite Zeile bei der Adresse 052B auf (Endadresse).

- b) Der ASCII-String für die zweite Zeile enthält $19_{10} = 13_{16}$ Bytes. Dies ist auch die Anzahl der Druckzeichen (einschließlich der Abstände mit dem Byte 20 – vgl. Seite T4 –) in der zweiten Zeile.

Sie müssen also folgende Tasten betätigen:

5, 1, 9, AD	1, 3,	0, 4,	PT
Anfangsadresse	Anz. der Zeichen	Druckart	Drucken

Zu Aufgabe S 147.1

Das Bild Ü 45.1 zeigt das Programm entsprechend dem Programmablaufplan im Bild S 147.1. Wir haben der Einfachheit halber für jeden Befehl nur eine Zeile benutzt.

Mit den ersten vier Befehlen wird das Pointer-Register P1 mit der Hilfsadresse 0800 geladen. Man hätte auch eins der Pointer-Register P2 oder

Nr.	Label	Operation	Operand	Adresse	Op-code	Displacement	Bemerkungen
1	ANFA	LDI	08	0420	C4	08	0800 → P1
2		XPAH 1		22	35		
3		LDI	00	23	C4	00	
4		XPAL 1		25	31		
5		LDI	C0	26	C4	C0	C0 → 0804
6		ST	04(P1)	28	C9	04	
7	BEGIN	LD	00(P1)	2A	C1	00	(0800)
8		ST	05(P1)	2C	C9	05	→ (0805)
9		JMP	BEG	2E	90	FA	

P3 für die Hilfsadresse wählen können. Die anschließenden Speicher- und Ladebefehle müssten dann nicht Pointer-1-indiziert, sondern Pointer-2- bzw. Pointer-3-indiziert programmiert werden.

Das Pointer-Register P3 wählt man allerdings zweckmäßig nicht. Bei unserem Mikroprozessor ist es üblich, dieses Pointer-Register für Adressen von Unterprogrammen zu reservieren. Wir halten es außerdem frei, um Programm-Halt-Anweisungen (BP) eingeben zu können.

Als Hilfsadresse erweist sich 0800 als vorteilhaft, weil sich dann die Displacements der Pointer-indizierten Befehle besonders leicht berechnen lassen. Bereits der Pointer-indizierte Befehl zum Speichern des Bytes C0 bei der Adresse 0804 zeigt das: Das Displacement errechnet sich zu

$$804_{16} - 800_{16} = 04_{16}$$

Die anschließenden Befehle LD 00(P1) und ST 05(P1) besorgen den Transport des Inhalts der Speicherzelle mit der Adresse 0800 in die Speicherzelle mit der Adresse 0804 auf dem Weg über den Akkumulator.

Als letzter wird der Sprungbefehl zum Label BEGIN bei der Adresse 042A programmiert. Da es sich nur um eine sehr kleine Sprungweite handelt, wählen wir einen PC-relativen Sprung. Zur Berechnung des Displacements setzen wir als Sprungziel die Adresse 0429 ein, die um genau eins kleiner ist als die Adresse 042A des nächsten, auszuführenden Befehls LD 00(P1) (Seiten S 66 und S 72). Startadresse ist die des Displacements des Sprungbefehls:

$$0429_{16} - 042F_{16} = FA_{16}$$

Bild Ü 45.1

Zu Aufgabe S 147.1. Dieses Programm stellt eine Verbindung zwischen dem Tastenfeld und der Sieben-Segment-Anzeige her.

Zu Aufgabe S 151.1

- a) Zum Ansteuern der (von links gezählt) ersten vier Stellen der Sieben-Segment-Anzeige müssen die Daten-Bits 4, 5, 6 und 7 die Werte 1 haben, die Daten-Bits 2 und 3 die Werte 0. Es ergibt sich folgendes Bitmuster (Bild S 149.1):

1111 00XX

Da die Daten-Bits 0 und 1 nicht zur Ansteuerung herangezogen werden, können sie beliebige Werte haben (X).

Dem angegebenen Bitmuster entsprechen die Bytes F0, F1, F2 und F3. Eins dieser Bytes muß anstelle des ursprünglich programmierten Bytes 10 bei der Adresse 042D ins Programm eingefügt werden. Machen Sie den Versuch!

- b) Jeder der Tasten ist ein Wert 1 an einer bestimmten Stelle des Bitmusters zugeordnet, das als Byte bei der Adresse 0800 abgeholt werden kann (Bilder S 150.1 und S 86.1. Werden mehrere Tasten gleichzeitig betätigt, dann erscheinen im Bitmuster mehrere Werte 1. Beim Betätigen der Tasten 0 und 2 ergibt sich z.B. das Bitmuster

0000 0101

Mit der Befehlsfolge LD 00(P1), XAE (Bild S 151.1) wird das entsprechende Byte 05 ins Extension-Register gebracht und anschließend mit dem Befehl LD 80(P2) der Inhalt der Speicherzelle mit der Adresse 0505 in den Akkumulator geladen. Es ist unwahrscheinlich, daß in der Speicherzelle mit der Adresse 0505 zufällig ein solches Byte steht, das dem Code für die Anzeige einer Ziffer (Bild S 150.2) entspricht.

- c) Das Byte 80, das beim Betätigen der Taste 7 bei der Adresse 0800 abgeholt wird, entspricht einer negativen Zahl: $80_{16} = -128_{10}$ (Bild S 73.2). — Nach den Befehlen LD 00(P1) und XAE in unserem Programm bringt die CPU den Inhalt 80_{16} des Extension-Registers automatisch als Displacement in den Ladebefehl bei der Adresse 0433 (Nr. 13 im Bild S 151.1). Es wird also der Inhalt der Speicherzelle mit der Adresse (P2) — $80_{16} = 0500_{16} - 80_{16} = 0480_{16}$ in den Akkumulator gebracht. In dieser Speicherzelle mit der Adresse 0480 müssen wir also den Code F8 zur Anzeige der Ziffer 7 ablegen.

a)

0500 01 23 45 67 89 AB CD EF

b)

0426 0500 0000 0000 00 10 00
 0426 0505 0000 0000 AB 10 00
 0426 0502 0000 0000 45 10 00
 0426 04FF 0000 0000 01 10 00

Zu Aufgabe S 157.1

Das Displacement $FD_{16} = -3_{10}$ (vgl. Bild S 73.2) im autoindizierten Ladebefehl bewirkt, daß der Inhalt des betreffenden Pointer-Registers vor dem Laden des Akkumulators um 3 erniedrigt wird.

Im Bild Ü 46.1a ist noch einmal der Inhalt des Stacks ausgedruckt. — Ausgehend von (P1) = 0508 wird mit der ersten Ausführung des Befehls LD @ FD(P1) zunächst (P1) = 0505 und dann der Inhalt AB der so indizierten Speicherzelle in den Akkumulator kopiert (Zeile 2 im Teilbild b). Bei der nächsten Ausführung des Befehls wird (P1) = 0502 und dann das Byte 45 geladen (Zeile 3).

Bei nochmaliger Ausführung des autoindizierten Ladebefehls wird (P1) = 04FF. Die Speicherzelle mit der Adresse 04FF gehört nicht mehr zu unserem Stack; es wird ein zufälliger Wert geladen. (In unserem Beispiel 01; Zeile 4.)

Zu Aufgabe S 161.1

Da nur die Inhalte von zwei Speicherzellen kopiert werden sollen, darf die Schleife im Programm nur zwei Mal durchlaufen werden. Der Zähler

muß also mit den Befehlen Nr. 9 und Nr. 10 und 02 gesetzt werden:
LDI 02; ST ZÄHL.

Das erste Byte, das aus dem Betriebsprogramm mit dem Befehl LD @1(P1) in den Akkumulator geholt werden soll, steht bei der Adresse 0019. Mit dieser Adresse muß P1 geladen werden. Sie brauchen nur die Lade-Anweisungen für das LOB dieses Pointer-Registers zu ändern: LDI 19, XPAL 1.

Ganz entsprechend müssen auch die Lade-Anweisungen für (P2) = 0519 modifiziert werden: LDI 19, XPAL 2.

Versuch Ü 47.1

Damit Sie die richtige Ausführung des geänderten Programms kontrollieren können, löschen Sie zweckmäßig die Speicher-Inhalte der Adresse 0501:

5, 0, 1, AD, IN, IN, IN ... (etwa 30 mal IN.)

Ändern Sie die Inhalte in den Lade-Anweisungen für P1, P2 und ZÄHL:
0424 19, 042A 19, 042D 02.

Starten Sie das Programm!

Schauen Sie sich den Inhalt des Speichers im Bereich zwischen den Adressen 0516 und 051D an! Sie können diese Inhalte auch ausdrucken lassen:

5, 1, 6, AD, 0, 8, 0, 2, PT:

0516 00 00 00 C1 01 00 00 00

Sie erkennen, daß tatsächlich nur die Inhalte der Speicherzellen mit den Adressen 0019 und 001A übertragen worden sind. (Vgl. Bild S 159.1.)

Zu Aufgabe H 164.1

Im Bild Ü 47.1 haben wir entsprechend der Darstellung im Bild H 163.1 das Bitmuster für die Segmentanschlüsse der Anzeige angegeben (linker Tabellenteil). Rechts sind die negierten Bits für die Segmentanschlüsse zusammengestellt, die als Byte 89₁₆ vom Datenbus an die Zweiweg-Bus-Treiber geliefert werden müssen.

Dieses Byte muß mit den Befehlen Nr.5 und Nr.6 (Bild Ü 45.1) an die Adresse 0804 geliefert werden.

Bild Ü 47.1
Zu Aufgabe H 164.1. Hier sind die Bitmuster zur Darstellung des Buchstaben H in der Sieben-Segment-Anzeige aufgelistet.

	$\overline{DB_7}$	$\overline{DB_6}$	$\overline{DB_5}$	$\overline{DB_4}$	$\overline{DB_3}$	$\overline{DB_2}$	$\overline{DB_1}$	$\overline{DB_0}$		DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0	Byte
Ziffer	DP	G	F	E	D	C	B	A		$\overline{DP}$	$\overline{G}$	$\overline{F}$	$\overline{E}$	$\overline{D}$	$\overline{C}$	$\overline{B}$	$\overline{A}$	
H	0	1	1	1	0	1	1	0		1	0	0	0	1	0	0	1	89

Prüfungsaufgaben Lehrbrief 7

Bitte beachten Sie beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü 1 und Ü 2.

1. Geben Sie bitte die Ergebnisse folgender Verknüpfungen in sedezimaler Form an:

a) $39_{16} \vee C6_{16}$	b) $A6_{16} \vee A8_{16}$	c) $14_{16} \vee 88_{16}$
d) $DE_{16} \wedge 26_{16}$	e) $59_{16} \wedge 38_{16}$	f) $21_{16} \wedge 4E_{16}$
g) $E4_{16} \oplus 9D_{16}$	h) $AB_{16} \oplus 54_{16}$	i) $3F_{16} \oplus 3F_{16}$

2. Die Teilaufgaben S 176.1b, c und d lassen sich auch mit einem Programm lösen, in dem nach dem Kopieren des Inhalts des Status-Registers in den Akkumulator zunächst das viermalige Rechts-Schieben programmiert und dann erst maskiert wird.

Im Bild Ü 49.1 sehen Sie den Ausdruck der entsprechenden Befehlsliste. Bei der Adresse 0425 muß der Befehl zum Maskieren stehen.

Geben Sie jeweils im mnemonischen Code und im Operationscode den Maskierungs-Befehl mit seinem Displacement für folgende Fälle an:

- a) SA wird nach F0 kopiert; SB bleibt unberücksichtigt.
 b) SB wird nach F1 kopiert; SA bleibt unberücksichtigt.
 c) SA wird nach F0 kopiert; SB wird nach F1 kopiert.

3. Lesen Sie bitte noch einmal beim Versuch S 184.1 nach! Wir sind dort davon ausgegangen, daß zunächst (ST) = 00 ist.

Sie sollen ein Programm schreiben, das – genau wie das Programm im Versuch S 184.1 – die negierten Werte von SA und SB bei den Bits F1 und F0 des Status-Registers erscheinen läßt. Zusätzlich sollen aber die Werte der Bits CY/L, OV, IE und F2 am Ende des Programms die gleichen Werte haben wie am Anfang des Programms. Ihre Werte sollen also von den Rechts-Schiebe- und Negations-Befehlen unbeeinflusst bleiben.

Im Bild Ü 50.1 haben wir ein Schema skizziert, mit dem diese Aufgabe gelöst werden kann. – Mit den Anweisungen 1 bis 3 wird (ST) → EX gerettet. Die Bits SA und SB werden maskiert (4) und an die Bitstellen 0 und 1 geschoben (5 bis 8). Nach dem Austausch (AC) ↔ (EX) (10) werden die Bits CY/L, OV, IE und F2 maskiert (11) und mit den jetzt verschobenen Bit SA und SB „verODERT“ (12). Zum Schluß wird das Bitmuster nach ST gebracht.

In unserem Bild haben wir die Bit-Stellen CY/L, OV, IE und F2, an denen die gespeicherten Werte unbeeinflusst bleiben sollen, durch rote Eintragungen X markiert. Die schwarz eingetragenen Werte 0 werden vom Programm gesetzt. Die jeweils fehlenden Eintragungen an einigen Bit-Stellen sind im Rahmen des Programms nicht von Interesse.

Im Programm müssen sämtliche Verknüpfungen verwendet werden, die der Befehlssatz unseres Mikroprozessors zuläßt.

Tragen Sie das zugehörige Programm in die Liste auf dem Lösungsblatt ein. Das Programm soll als Schleife laufen!

```
0420 05
0421 1C
0422 1C
0423 1C
0424 1C
0425 XXYY
0427 07
0428 90F6
```

Bild Ü 49.1

Zu Prüfungsaufgabe 2: Welche zwei Bytes müssen anstelle der roten Eintragungen in dieser Liste stehen, damit die im Aufgabentext genannten Bedingungen erfüllt werden?

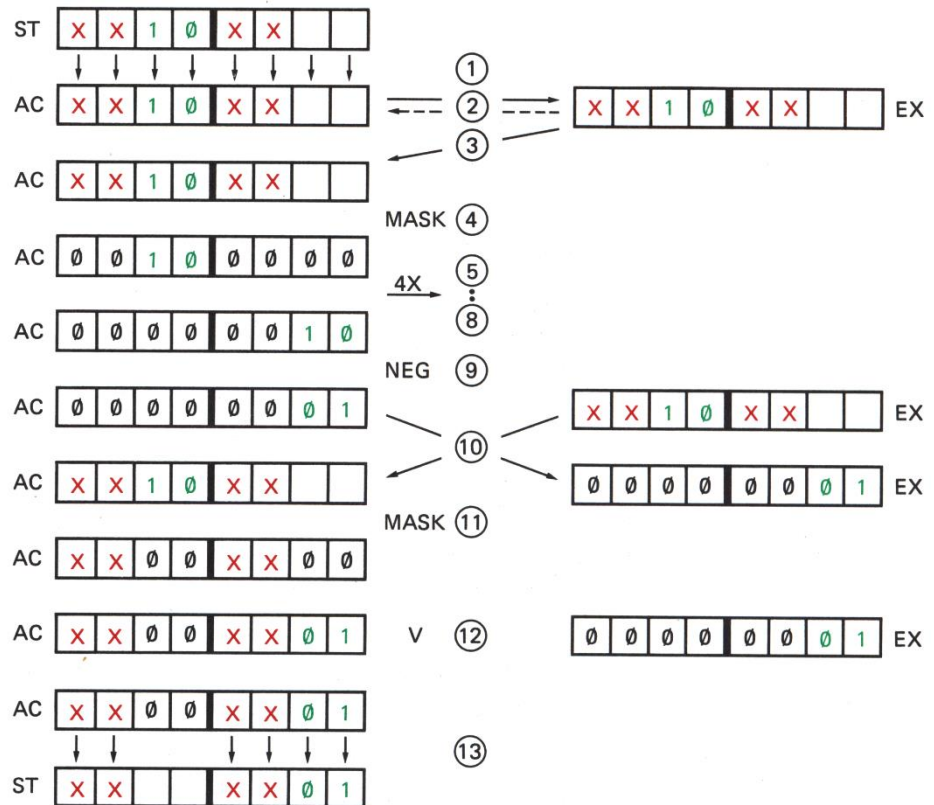


Bild Ü 50.1

Zu Prüfungsaufgabe 3: Nach diesem Schema können die negierten Bits SA und SB an die Bit-Stellen F0 und F1 kopiert werden, ohne daß die Werte der Bits CY/L, OV, IE und F2 beeinflusst werden.

4. Diese Aufgabe bezieht sich auf den Text auf der Seite S 193: Zur Abfrage von „Eingabe-Stationen“ laufen viele Programme in einer Warteschleife.

Geben Sie bitte an, welche Anweisungen im Programmablaufplan im Bild H 170.1 zur Abfrage-Schleife für das Tastenfeld gehören!

5. Im Würfelspiel-Programm (Seiten S 207 und S 208) wird das Würfel-Ergebnis an der 16^0 -Stelle der Operationscodes angezeigt.

Welche (sehr einfache!) Änderung muß im Programm vorgenommen werden, wenn das Würfel-Ergebnis an der 16^0 -Stelle der Adresse angezeigt werden soll?

Geben Sie für die Änderung die Befehlsnummer, die Operation, den Operanden, die Adresse sowie den Opcode und das Displacement auf dem Lösungsblatt an!

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe H 186.1

An den Speicherausgängen erscheinen nacheinander die Datenworte 08, C4, 04, 37, C4, FF, 33, 3F.

Als Befehle für unseren Mikroprozessor interpretiert ergeben diese Datenworte die Befehlsfolge

NOP, LDI 04, XPAH 3, LDI FF, XPAL 3, XPPC 3. (Seite T 8)

Nach einem Leerbefehl (keine Operation) wird von dieser Befehlsfolge das Pointer-Register P3 mit 04FF₁₆ geladen. Anschließend wird dieses Zwei-Byte-Wort in den Programmzähler (PC) gebracht. Der nächste, auszuführende Befehl steht dann bei der Adresse 0500.

Zu Aufgabe S 171.1

Das Bild Ü 51.1 zeigt die Liste des geänderten Programms. Als wichtigste Änderung ist zu erkennen, daß nach dem Verknüpfungsbefehl jetzt drei Rechts-Schiebe-Befehle folgen.

Das Ergebnis der Verknüpfung wird nun von der 2⁴-Stelle an die 2¹-Stelle geschoben und erscheint deshalb nach dem CAS-Befehl im Status-Register an der Stelle des Bits F1. Die Lumineszenzdiode F1 zeigt das Ergebnis der Verknüpfung an.

Da jetzt das Maskieren und Invertieren an der 2¹-Stelle vorgenommen werden muß, erhalten die Befehle ANI und XRI die Displacements 02.

Nr.	Label	Operation	Operand	Adresse	Op-code	Displacement	Bemerkungen
1	BEG	CSA		0420	06		(ST) → AC
2		XAE		21	01		(AC) → EX
3		CSA		22	06		(ST) → AC
4		RR		23	1E		1 x RECHTS SCHIEBEN
5		ANE		24	50		(AC) ∧ (EX) → AC
6		RR		25	1E		3 x RECHTS SCHIEBEN
7		RR		26	1E		
8		RR		27	1E		
9		ANI	02	28	D4	02	BIT 2 MASKIEREN
10		XRI	02	2A	E4	02	BIT 2 INVERTIEREN
11		CAS		2C	07		(AC) → ST
12		JMP	BEG	2D	90	F1	SCHLEIFE

Bild Ü 51.1

Zu Aufgabe S 171.1. Durch den zusätzlich eingefügten RR-Befehl wird das Verknüpfungsergebnis von der Lumineszenzdiode F1 angezeigt.

Zu Aufgabe S 174.1

- a) Das erste der beiden miteinander zu verknüpfenden Bytes wird mit dem Sofort-Befehl in den Akkumulator geladen und dann per Austausch-

Bild Ü 52.1

Zu Aufgabe S 174.1b. Nach Ablauf dieser vier Befehle steht das Ergebnis der Verknüpfung $(AC) \wedge (EX)$ im Akkumulator.

Operation	Operand	Adresse	Op-code	Displacement	Bemerkungen
LDI	DA	0420	C4	DA	DA → AC
XAE		22	01		DA → EX
LDI	6D	23	C4	6D	6D → AC
ANE		25	50		DA ∧ 6D → AC

Befehl ins Extension-Register gebracht. Anschließend wird das zweite Byte mit dem Sofort-Befehl in den Akkumulator geladen. — Der ANE-Befehl besorgt die UND-Verknüpfung $(AC) \wedge (EX) \rightarrow AC$.

b) Bild Ü 52.1

Versuch Ü 52.1

Die UND-Verknüpfung von zwei Bytes

Tasten Sie die im Bild Ü 52.1 aufgelistete Befehlsfolge ein und betätigen Sie bei der Adresse 0426 die Taste BP! — Starten Sie das Programm (GO)! Anschließend können Sie das Verknüpfungsergebnis als Akkumulator-Inhalt mit der Taste AC abrufen.

7	6	5	4	3	2	1	0
0	1	1	0	1	0	0	1

Bild Ü 52.2

Zu Aufgabe S 176.1a. In diesem Bitmuster des Displacements bewirken die grün eingetragenen Werte 1 die Maskierung der Bits mit den Nummern 0, 3, 5 und 6.

Zu Aufgabe S 175.1

- a) 01_{16} b) 10_{16} c) $A8_{16}$ d) 20_{16}

Zu Aufgabe S 176.1

- a) Im Bild Ü 52.2 haben wir die Öffnungen der Maske mit grün eingetragenen Werten 1 gekennzeichnet. Für das Displacement des ANI-Befehls ergibt sich das Byte 69_{16} .
- b) Im Bild 176.2 ist bereits angedeutet, daß die Befehlsfolge in einer Schleife abgearbeitet werden muß, damit eine kontinuierliche Übertragung des Bits SA an die Stelle des Bits F0 stattfindet.

Bild Ü 52.3

Zu Aufgabe S 176.1b. Mit dieser Befehlsfolge wird das Bit SA außer von der Lumineszenzdiode SA auch von der Lumineszenzdiode F0 zur Anzeige gebracht. Das Bit SB wird nur von der Lumineszenzdiode SB angezeigt.

Das Bild Ü 52.3 zeigt die Befehlsfolge, die das gewünschte Verhalten unseres Systems verursacht: Der Inhalt des Status-Registers wird in den Akkumulator kopiert; dort wird das Bit Nr. 4 (SA) maskiert. Mit den anschließenden vier SR-Befehlen wird das gesamte Bitmuster, das nun höchstens einen Wert 1 enthält, so verschoben, daß der eventuell vor-

Label	Operation	Operand	Adresse	Op-code	Displacement	Bemerkungen
BEGIN	CSA		0420	06		(ST) → AC
	ANI	10	21	D4	10	MASKE FÜR SA
	SR		23	1C		4 x RECHTS SCHIEBEN
	SR		24	1C		
	SR		25	1C		
	SR		26	1C		
	CAS		27	07		(AC) → ST
	JMP	BEGIN	28	90	F6	SCHLEIFE

handene Wert 1 zum Bit Nr. 0 (F0) wird. Durch das Kopieren des Akkumulator-Inhalts ins Status-Register wird der anfangs bei SA eingegebene Wert bei F0 zu Anzeige gebracht.

- c) Die Struktur der Befehlsfolge bleibt unverändert. Lediglich die Maske wird geändert: Das im Bild Ü 52.3 grün eingetragene Byte wird auf den Wert 20_{16} gesetzt. Dadurch wird das Bit SB maskiert. Wenn bei laufendem Programm über den Tastschalter SENSE B das Bit Nr. 5 (SB) auf den Wert 1 gesetzt wird, dann wird mit den vier SR-Befehlen der Wert 1 zum Bit Nr. 1 geschoben. Nach Ausführung des CAS-Befehls leuchtet die Lumineszenzdiode F1.

Der Wert SA = 1 beim Betätigen des Tastschalters SENSE A wird wegmaskiert; die Lumineszenzdiode F0 leuchtet nicht.

- d) Die Struktur der Befehlsfolge bleibt auch jetzt unverändert. Das im Bild Ü 52.3 grün eingetragene Byte wird auf 30_{16} geändert. Die Maske hat dann solche Öffnungen, daß sowohl das Bit SA als auch das Bit SB wirksam bleiben.

Sie können sich von der Richtigkeit der Lösungen b, c und d im Versuch überzeugen.

Zu Aufgabe S 183.1

- a) Im Bild Ü 53.1 haben wir den Ablaufplan für das im Bild S 180.2 aufgelistete Programm noch einmal in etwas modifizierter Form dargestellt. Die zu den einzelnen Befehlen gehörenden Mikrozyklen finden Sie auf der Seite T 8 aufgelistet. Danach ergibt sich für die Lade-Anweisung für ein Pointer-Register, bestehend aus den Befehlen LDI, XPAH, LDI, XPAL, die Summe von 36 Mikrozyklen. (Rote Eintragung neben der Anweisung 1 im Bild Ü 53.1.)

Bei der Anweisung 5 kommen zu den 36 Mikrozyklen zum Laden des Pointer-Registers P2 noch 18 Mikrozyklen für den ST-ZÄHL (P1)-Befehl.

Bis zum Label HOCH braucht das Programm demnach $(36 + 54) \text{ Mz} = 90 \text{ Mz}$.

Für einen vollständigen Schleifenumlauf, bestehend aus den Befehlen ILD, XOR und dem ausgeführten, bedingten Sprung JNZ braucht das Programm insgesamt 51 Mikrozyklen. Für fünfmaliges Inkrementieren des Zählers in der Schleife werden also $5 \cdot 51$ Mikrozyklen benötigt, von denen jedoch zwei Mikrozyklen abgezogen werden müssen, da der bedingte Sprung nach dem letzten Schleifendurchlauf nicht mehr ausgeführt wird.

Für die Schleifendurchläufe werden also $(5 \cdot 51 - 2) \text{ Mz} = 253 \text{ Mz}$ benötigt.

Für den Rücksprung zum Anfang des Betriebsprogramms werden 11 Mz gebraucht.

Das Programm braucht demnach $(90 + 253 + 11) \text{ Mz} = 354 \text{ Mz}$.

Da bei unserem System $1 \text{ Mz} \approx 1,91 \mu\text{s}$ ist, beträgt die Ablaufdauer des Programms

$$354 \cdot 1,91 \mu\text{s} \approx 676 \mu\text{s}$$

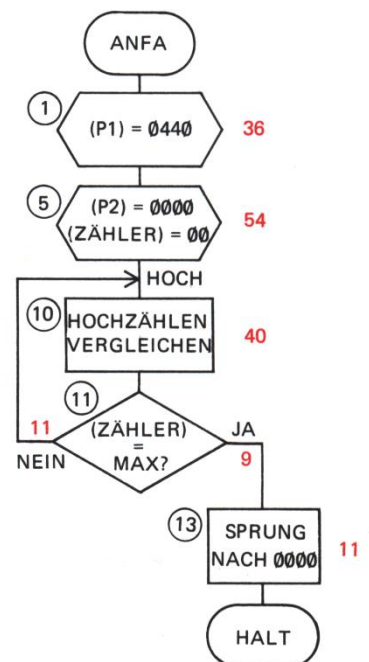


Bild Ü 53.1

Zu Aufgabe S 183.1. Hier ist der Programmablaufplan aus dem Bild S 180.1 so modifiziert dargestellt, daß wir die zu den Anweisungen gehörende Anzahl der Mikrozyklen eintragen konnten.

b) Bis zum Label HOCH benötigt das Programm 90 Mz.

Beim Hochzählen bis (ZÄHL) = FF muß die Schleife 255 mal durchlaufen werden. Da nach dem letzten Schleifendurchlauf der Sprungbefehl JNZ nicht mehr ausgeführt wird, werden zum Hochzählen $(255 \cdot 51 - 2)$ Mz = 13 003 Mz benötigt.

Für den Rücksprung zum Anfang des Betriebsprogramms braucht das Programm 11 Mz.

Insgesamt werden für das Programm $(90 + 13\,003 + 11)$ Mz = 13 104 Mz gebraucht. Das entspricht einer Ablaufdauer von $13\,104 \cdot 1,91 \mu\text{s} \approx 25\,029 \mu\text{s}$.

Zu Aufgabe S 185.1

Im Displacement des XRI-Befehls muß das an der Stelle des Bits SA des Status-Registers stehende Bit Nr. 4 den Wert 0 haben und das an der Stelle des Bits SB des Status-Registers stehende Bit Nr. 5 den Wert 1 haben.

Da wir voraussetzen, daß außer den Bits SA und SB im Status-Register alle übrigen Bits mit Sicherheit die Werte 0 haben, müssen nach der Ausführung des XRI-Befehls diese Bits unverändert mit den Werten 0 im Akkumulator stehen. Im Displacement des XRI-Befehls müssen also auch die Bits 0, 1, 2, 3, 6 und 7 die Werte Null haben.

Für das Displacement des XRI-Befehls ergibt sich also das Bitmuster

$$0\,0\,1\,0\,0\,0\,0\,0 = 20_{16}$$

Zu Aufgabe S 186.1

Der Fehler im Bild S 186.2 ist nur im Zusammenhang mit der Programmliste im Bild S 151.1 erkennbar. Es handelt sich um einen Fehler, der auch einem geübten Programmierer hin und wieder im Eifer des Gefechts unterläuft, wenn er sich nicht eines Assemblierers (Seite S 162) bedient: Es wurde vergessen, das Displacement des Befehls JMP BEGIN zu ändern.

Durch den zusätzlich eingefügten Befehl ist die Weite des Rücksprungs größer geworden. Das Bild Ü 54.1 enthält in grüner Eintragung den richtigen Wert des Displacements für den Sprung zum Label BEGIN.

Bild Ü 54.1

Zu Aufgabe S 186.1. Das im Bild S 186.1 falsch eingetragene Displacement der Rücksprung-Anweisung haben wir hier mit dem richtigen Wert grün eingetragen.

Ü

54

Nr.	Label	Operation	Operand	Adresse	Op-code	Displacement	Bemerkungen
13		LD	80 (P2)	0433	C2	80	(05 [EX])
13a		XRI	FF	35	E4	FF	(AC) NEGIEREN
14		ST	04 (P1)	37	C9	04	→ 0804
15		JMP	BEGIN	39	90	F5	

Zu Aufgabe H 198.1

a) Nach dem Starten des Programms leuchten die Lumineszenzdioden CY/L, OV und IE auf.

Mit dem ersten Speicherbefehl (Nr. 6 im Bild H 196.2) wird in den Flipflops im IC 0807 das Bitmuster

7	6	5	4	3	2	1	0	<i>Datenbit</i>
1	1	1	1	1	0	—	—	

gespeichert. — Die hier grün eingetragenen Bits dienen zur Steuerung der Relais. Der Schaltplan im Bild H 193.2 zeigt, daß die Ausgänge der zu den Bits mit den Nummern 7, 6 und 3 gehörenden Flipflops über Widerstände mit den Lumineszenzdioden CY/L, OV und IE verbunden sind. Die n-Anschlüsse dieser Lumineszenzdioden sind mit dem Potential 0 V verbunden.

Die Werte 1 der Bits mit den Nummern 7, 6 und 3 bringen die genannten Lumineszenzdioden also zwangsläufig zum Leuchten.

Auch nach den folgenden Speicherbefehlen haben die Bits mit den Nummern 7, 6 und 3 an den Ausgängen der hier betrachteten Flipflops stets die Werte 1. Die Lumineszenzdioden CY/L, OV und IE werden also dauernd leuchten.

- b) Die Bits mit den Nummern 7, 6 und 3 an den Ausgängen der Flipflops müssen nach jedem der Speicherbefehle die Werte 0 haben. Der erste Speicherbefehl (Nr. 6), der das Ansprechen des Relais Ry 6 verursacht, muß also folgendes Bitmuster auf den Bus mit den negierten Daten bringen:

7	6	5	4	3	2	1	0	<i>Datenbit</i>
0	0	X	1	0	0	X	X	

Die Bits mit den Nummern 1 und 0 werden nicht gespeichert; das gespeicherte Bit Nr. 5 bleibt ungenutzt (Bild H 193.2). Für diese Bits können wir also Don't-care-Werte X einsetzen (0 oder 1, nach Belieben) — Wegen der negierenden Zweiweg-Bus-Treiber muß der Speicherbefehl das invertierte Bitmuster transportieren. Das Bitmuster des Displacements des vorhergehenden Sofort-Ladebefehls LDI (Nr. 5) hat also folgende Form:

1	1	X	0	1	1	X	X
---	---	---	---	---	---	---	---

Sedezimal entspricht dem z.B. das Byte CC, wenn wir X = 0 einsetzen.

Der Befehl Nr. 5 muß demnach z.B. folgende Form erhalten:

LDI CC 0426 C4 CC

Für die Sofort-Ladebefehle 8, 11 und 14 gelten entsprechende Überlegungen. Wir schreiben die Bitmuster hier ohne weiteren Kommentar der Reihe nach an, links für den Bus mit den negierten Daten und rechts für den von der CPU kommenden Datenbus von den Zweiweg-Bus-Treibern:

7	6	5	4	3	2	1	0	<i>Datenbit</i>
0	0	X	0	0	0	X	X	
0	0	X	0	0	1	X	X	
0	0	X	1	0	1	X	X	

7	6	5	4	3	2	1	0	<i>Datenbit</i>
1	1	X	1	1	1	X	X	
1	1	X	1	1	0	X	X	
1	1	X	0	1	0	X	X	

Außer der Änderung des Befehls Nr. 5 müssen also folgende Änderungen vorgenommen werden: (Wir setzen wieder $X = 0$.)

```

8 LDI DC    042C C4 DC
11 LDI D8    0432 C4 D8
14 LDI C8    0438 C4 C8

```

Zu Aufgabe S 189.1

Ist Ihnen aufgefallen, daß wir in der Aufgabenstellung von einem Sonderprogramm und nicht von einem Unterprogramm gesprochen haben?

Ein Unterprogramm muß mehrfach aufgerufen werden können (Seite S6). Am Ende eines solchen Programms muß immer ein Rücksprung-Befehl zu der Stelle des Hauptprogramms stehen, an der es aufgerufen wurde. Weil ein Unterprogramm innerhalb des Hauptprogramms mehrfach, also an verschiedenen Stellen, aufgerufen werden kann, muß der Rücksprung-Befehl variabel sein. Das ist bei unserem Mikroprozessor nur bei Verwendung des XPPC-Befehls möglich. (Seite S 66.)

Bei der Bearbeitung eines Interrupts muß es sich um ein echtes, an verschiedenen Stellen des Hauptprogramms aufrufbares Unterprogramm handeln. Beim direkten Ansprung des „Sonderprogramms“ mit einem bedingten Sprungbefehl würde die Aussprung-Adresse aus dem Hauptprogramm nicht aufbewahrt („gerettet“).

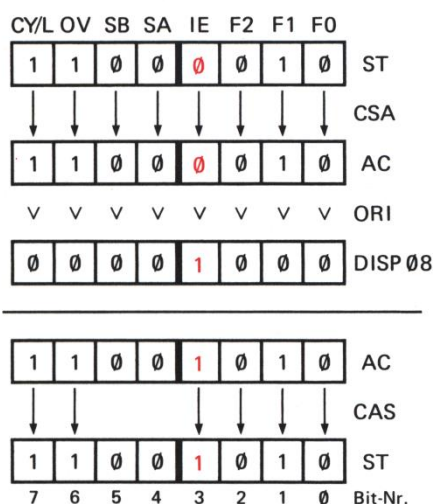


Bild Ü 56.1

Zu Aufgabe S 195.1. Nach diesem Schema kann das Bit *IE* im Status-Register gesetzt werden, ohne daß die Werte der übrigen Bits beeinflusst werden.

Zu Aufgabe S 195.1

Zur Lösung dieser speziellen Aufgabe gibt es im Befehlssatz unseres Mikroprozessors einen gesonderten Befehl.— Ohne diesen Befehl ließe sich die Aufgabe nur unter Verwendung des ODER-Verknüpfungsbefehls lösen.

Der Inhalt des Status-Registers muß in den Akkumulator kopiert werden. Dann besorgt der Sofort-ODER-Befehl (Seite S 177) mit dem Displacement 08 das 1-Setzen des Bits Nr. 3, ohne die Werte der übrigen Bits zu verändern. Anschließend wird das Byte aus dem Akkumulator ins Status-Register zurück-kopiert. Es ergibt sich also diese Befehlsfolge:

CSA, ORI 08, CAS

Das Bild Ü 56.1 zeigt anhand der Bitmuster die entsprechenden Vorgänge.

Zu Aufgabe S 199.1

Nichts passiert! Versuchen Sie es und ersetzen Sie bei der Adresse 0466 den Befehl IEN durch den Befehl NOP! Das Programm läuft wie vorher. Warum?

Bitte, das ist in unserem Programm nur zufällig so, wie es auf der Seite S 198 für den Befehl IEN bei der Adresse 0426 beschrieben wurde! Sonst muß IEN am Ende einer Interrupt-Routine unbedingt programmiert werden!

Prüfungsaufgaben Lehrbrief 8

Bitte beachten Sie beim Anfertigen und Einsenden der Lösungen zu diesen Prüfungsaufgaben die allgemeinen Hinweise auf den Seiten Ü1 und Ü2.

1. Im Versuch S 215.1 führt das Mikroprozessor-System mit dem im Bild S 216.1 aufgelisteten Programm die Addition

$$\begin{array}{r}
 5 \ 1 \ 7 \ A_{16} \\
 + \quad A \ F \ B \ 4_{16} \\
 \hline
 (\emptyset) \ \textcolor{red}{1} \ 0 \ 1 \ 2 \ E_{16}
 \end{array}$$

aus. Der hier rot gedruckte Übertrag 1 an der 16^4 -Stelle ergibt sich aus dem Wert $CY/L = 1$ bei der Addition der HOBs.

Führen Sie den Versuch noch einmal durch!

Warum leuchtet die Lumineszenzdiode CY/L beim Ablauf des Programms nicht auf?

(Lesen Sie noch einmal den Abschnitt über die Bitmuster-Anzeige ab Seite H 193 durch!)

2. Mit dem Befehl SRL kann das Status-Bit CY/L von links nach rechts in den Akkumulator geschoben werden. Mit jedem SRL-Befehl wird das Bitmuster im Akkumulator um eine Bitstelle nach rechts geschoben. (Vgl. Seite S 239.)

Es läge nahe, im Programm im Bild S 210.1 die Befehle Nr. 19 (CSA) und Nr. 20 bis Nr. 26 (siebenmal SR) durch insgesamt acht SRL-Befehle zu ersetzen.

Machen Sie einen entsprechenden Versuch mit der Addition $89_{16} + 7E_{16}$ und geben Sie an, wie der in unserem Programm nicht sinnvolle Ausdruck zustande kommt!

3. Bei der parallel-Serie-Umwandlung der Bytes in einem String im Versuch S 247.1 haben wir nicht vorgesehen, die Länge des Strings zu begrenzen. Es werden also im Speicher aufeinanderfolgende Bytes solange bitseriell ausgegeben, bis das Programm durch Betätigen der Taste RESET angehalten wird.

Eine programmierte Vorgabe der Anzahl der bitseriell auszugebenden Bytes ließe sich durch einen zusätzlichen Zähler erreichen, der nach jedem gewandelten Byte dekrementiert wird. Dazu müßte aber am Anfang des Programms eine Anweisung zum Setzen des Zählers programmiert werden.

Auf dem Lösungsblatt am Ende des Lehrbriefs haben wir einige Befehle in mnemonischen Codes angeschrieben, die dem im Bild S 244.1 aufgelisteten Programm als „Rucksack“ angehängt werden können. Mit

diesen Befehlen wird geprüft, ob der Inhalt des Pointer-Registers P1, der bei der Anweisung Nr. 9 (Bilder S 244.1 und S 246.1) jeweils inkrementiert wird, bereits einen vorgegebenen Wert erreicht hat oder nicht.

Der Beginn des Strings mit den bit-seriell auszugebenden Bytes wird in der Anweisung Nr. 1 durch Setzen des Pointer-Registers P1 programmiert. Das Ende des Strings wird im Rucksack mit dem Displacement des Befehls Nr. 48 [LDI LOB (END + 1)] angegeben. Anschließend wird per Programm untersucht, ob das LOB des Inhalts des Pointer-Registers P1 den vorgegebenen Wert erreicht hat. Ist das der Fall, dann springt das Programm an den Anfang des Betriebsprogramms und bleibt dann bei der Adresse 0420 stehen.

Assemblieren Sie bitte das auf dem Lösungsblatt angegebene Programm! Mit anderen Worten: Ergänzen Sie die Adressen und tragen Sie die Operationscodes und Displacements ein!

Geben Sie vor allem in der ersten Zeile der Programmliste an, welcher Befehl des Hauptprogramms (Bild S 244.1) in einen Sprung zum Rucksack geändert werden muß. Tragen Sie die Befehls-Nummer, die Adresse, den neuen Operationscode und das Displacement ein! (Das Displacement des Befehls Nr. 50 gibt Ihnen einen Hinweis!)

4. Welchen Wert nimmt das Status-Bit OV nach Ausführung folgender Additionen in unserem Mikroprozessor an:

a) $7D_{16} + 48_{16}$ b) $3F_{16} + 29_{16}$

Lösungen der im Text gestellten Aufgaben

Bitte sehen Sie sich die folgenden Lösungen erst dann an, wenn Sie die im Text gestellten Aufgaben selbstständig durchgearbeitet haben.

Zu Aufgabe H 217.1

Die größte dual mit sechzehn Stellen angebbare Zahl ist $1111\ 1111\ 1111\ 1111_2 = \text{FFFF}_{16} = 65535_{10}$. Dies ist die höchste Adresse, die auf unserem erweiterten Adreßbus angegeben werden kann.

Da auch $0000\ 0000\ 0000\ 0000_2 = 0000_{16}$ eine verwendbare Adresse ist, können mit unserem Mikroprozessor 65536_{10} Speicherzellen eindeutig adressiert werden.

Dem entspräche ein Speicher mit 64 KByte. (Seite H 120.)

Zu Aufgabe S 217.1

Im Bild Ü 59.1 haben wir das Schema dargestellt, nach dem die Aufgabe in unserem Programm gelöst wird. Wir haben die den farbigen Unterlegungen im Bild S 216.1 entsprechenden Farben für die Ziffern gewählt. In allgemeinen Zahlen, also in Buchstaben, sieht die Additionsaufgabe so aus:

O P Q R + S T U V

Daraus ergibt sich für unsere Aufgabe folgende Zuordnung

Adresse 0435 OP = 2 E Adresse 0439 ST = F7
Adresse 0441 QR = A5 Adresse 0445 UV = 5 C

Nach dem Einsetzen dieser Werte und Start des Programms liefert der Drucker (in entsprechender Lesart) folgenden Ausdruck :

0400 A5 2E + 5C F7 = 01 02 25

Zu Aufgabe S 217.2

Vor der erstmaligen Anwendung eines Additionsbefehls wird ein CCL-Befehl (Seite S 215) programmiert, der das CY/L-Bit gezielt zu null macht. Diese Maßnahme empfiehlt sich grundsätzlich, wenn vor einem Additionsprogramm noch andere Programmteile laufen.

Zu Aufgabe S 220.1

Die Pointer-2-indizierten ADD-Befehle müssen durch ebenfalls Pointer-2-indizierte DAD-Befehle ersetzt werden (Seite S 219).

Zum Umstellen des Programms müssen also folgende Änderungen vorgenommen werden:

043C EA; 0448 EA

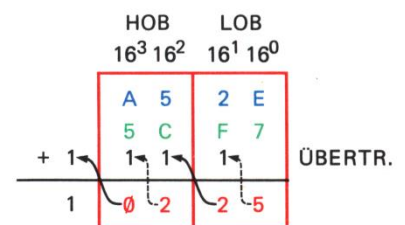


Bild Ü 59.1
Zu Aufgabe S 217.1: Schema zur Lösung der Aufgabe AS2E₁₆ + 5CF7₁₆.

Zu Aufgabe S 224.1

- a) 0 1 0 1 1 1 0 0 Invertiertes Bitmuster des Bytes A3₁₆.
 0 1 0 1 1 1 0 1 Bitmuster des Zweierkomplements des Bytes A3₁₆.
- b) 1 0 1 0 0 0 0 1 Invertiertes Bitmuster des Bytes 5E₁₆.
 1 0 1 0 0 0 1 0 Bitmuster des Zweierkomplements des Bytes 5E₁₆.
- c) 0 0 0 0 1 1 1 0 Invertiertes Bitmuster des Bytes F1₁₆.
 0 0 0 0 1 1 1 1 Bitmuster des Zweierkomplements des Bytes F1₁₆.

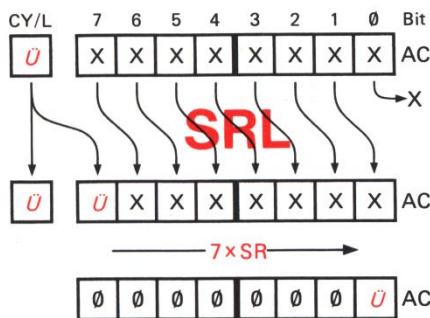
Das Bitmuster des Zweierkomplements ergibt sich aus der Addition des Werts 1 zum invertierten Bitmuster.

Zu Aufgabe S 233.1

Nach dem Befehl Nr.19 ADI 9A im Programm im Bild S 231.1 steht im Akkumulator das Hunderter-Komplement des Subtrahenden.

Geben Sie den Minuenden und den Subtrahenden jeweils wie folgt ein:

- a) Adresse 0436 $TU = 47$; Adresse 043 A $VW = 16$
 b) Adresse 0436 $TU = 89$; Adresse 043 A $VW = 23$
 c) Adresse 0436 $TU = 95$; Adresse 043 A $VW = 73$



Lassen Sie nach der jeweiligen Eingabe das Programm in Einzelschritten ablaufen, bis die Adresse 0441 erscheint. Beim Betätigen der Taste AC werden dann folgende Hunderter-Komplemente der Subtrahenden angezeigt:

- a) (AC) = 84 b) (AC) = 77 c) (AC) = 27

Zu Aufgabe S 240.1

Der Befehl SRL kann in diesem Fall den Befehl CSA ersetzen. Das Bild Ü 60.1 zeigt schematisch den Weg, den das mit $\bar{U}$ bezeichnete Übertrags-Bit vom Status-Bit CY/L zur Bit-Stelle Nr.0 im Akkumulator nimmt.

Für die Beurteilung eines Programms ist die für das Erreichen eines bestimmten Ziels benötigte Anzahl von Speicherzellen für Befehls-Bytes und die Anzahl der Mikrozyklen beim Ablauf dieses Programms maßgeblich. Je weniger Speicherzellen und je weniger Mikrozyklen man im Programm braucht, um so besser ist es.

Im Programm entsprechend dem Bild S 210.1 bleibt bei Verwendung des Befehls SRL die Anzahl der benötigten Speicherzellen gleich.— Da sowohl der Befehl CSA als auch der Befehl SRL fünf Mikrozyklen benötigen, läuft das Programm in beiden Fällen gleich schnell ab. Die Programme sind also in beiden Fällen genau gleichwertig.

Zu Aufgabe S 248.1

Die Schiebe-Geschwindigkeit wird in unserem Programm durch die Anweisungen Nr.25 und Nr.33 bestimmt. Wenn die Schiebe-Geschwindigkeit größer werden soll, dann müssen kleiner Verzögerungen programmiert werden.

Bild Ü 60.1

Zu Aufgabe S 240.1: Das Übertrags-Bit $CY/L = \bar{U}$ kann mit dem Befehl SRL an die Bitstelle Nr. 7 im Akkumulator gebracht werden.