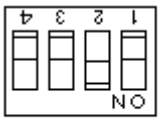
MYCPU80操作説明書

〒463-0067　名古屋市守山区守山2－8－14
パレス守山305
有限会社中日電工
TEL052-791-6254　Fax052-791-1391
E-mail　thisida@alles. or. jp

[注意]この説明書はMYCPU80組立説明書にしたがって組み立ての途中で動作テストを行うことを前提にして書かれています。組立説明書Ⅲ[13]TK80回路を組み立てたあとではディップスイッチやショートストラップの設定が異なってきます。

もしTK80回路を組み立てたあとで、この説明書に記載の動作テストを行う場合には、あらかじめデイップスイッチDS3の2を図のように下側（ON）にしておいてください。



DS3

またショートストラップSP1、SP2のショートピンは全部外してください。

なお上記のようにしないでTK80の回路を有効にしたままでも、トグルスイッチを使ってメモリにデータを書いたり、メモリの値を読んだりすることはできますが、メモリアドレスの0000～7FFFはROMに割り当てられているため、その範囲にプログラムやデータを書き込むことはできません。

またリセット後はTK80モニタのROMがアクセスされるため、アドレス8000以後のRAMエリアにプログラムを書いても、この説明書で説明する方法でのテストは行えません。

目次

2009. 9. 19 Rev. 1. 1（2f）
2009. 10. 11 Rev. 1. 2（2i）

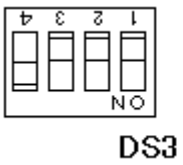
1.　メモリにデータを書き込む

この1. と次の2. は、MYCPU80組立説明書　Ⅲ組立　[2]メモリ(RAM)回路とスイッチ入力回路　の組立作業後に行う動作テストの説明です。

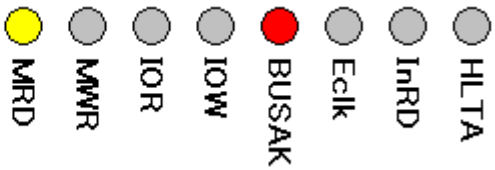
最初はメモリの書き込み、読み出しテストを行うだけですから、適当なデータをメモリに書き込んでみます。
アドレス0000に12を、0001に34を、0002に56を、0003に78をそれぞれ書き込んでみます(数値は全て16進数です)。

1－1.　ディップスイッチ(DS3)のセット

トグルスイッチを使ってメモリに直接データ(プログラムも同じ)を書き込むときは、ディップスイッチDS3の一番左側の4のみを、小型のマイナスドライバを使って図のようにON(下側)にセットします。



DS3



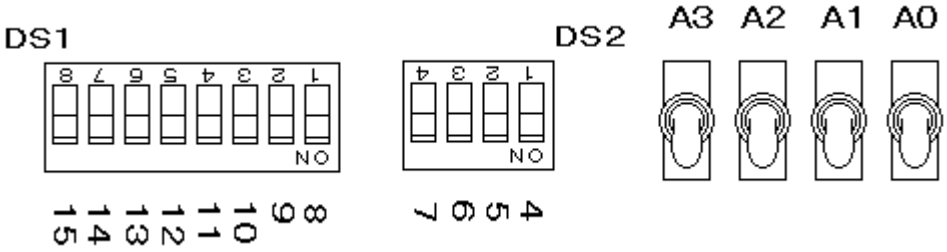
DS3－4をONにすると、アドレスバス、データバスやその他の制御信号ラインがCPUから切り離されて、外部から直接メモリにアクセスすることができるようになります。
このスイッチ(DS3の4番)は外部からCPUにバス信号線を開放することを要求するBUSRQ(BUS　Request)につながっていて、スイッチをONにするとBUSRQがアクティブになって、CPUから出力されているBUS信号線やその他の制御信号線がハイインピーダンスになり、外部から制御できるようになります。
BUSが開放されたことを示すLED(BUSAK)が点灯します。同時にメモリがREADアクセス状態になるため、MRD(Memory　Read)も点灯します。

[LEDの色]
MYCPU80基板に実装しているLEDは一般的な赤色のほかに、黄色と緑色のものがあります。
例外もありますが、基本的に黄色はREAD関係、緑色はWRITE関係の制御信号に使っています。

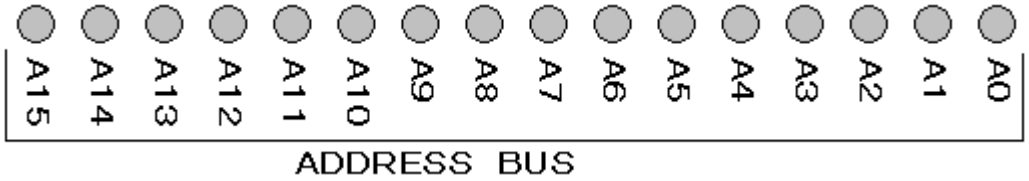
1－2.　メモリアドレスのセット

ディップスイッチのDS1、DS2とトグルスイッチのA3～A0を使って、メモリアドレスをセットします。
この2つのディップスイッチと12個のトグルスイッチはBUSRQがアクティブのときだけ有効になります。
回路図№1でDS3－4をONにすると、その信号が回路図№30のT1(トランジスタC1815)に送られて、T1がアクティブになることで、アドレススイッチ、データスイッチがGNDとつながります。

図のようにディップスイッチのDS1、DS2とトグルスイッチのA3～A0の全部を下側にします。スイッチを上側にすると"1"、下側にすると"0"がセットされます。

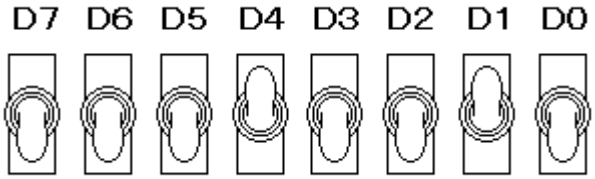


これでアドレスが2進数表示の0000　0000　0000　0000(16進数表示の0000)にセットされたことになります。
ディップスイッチは小さいので、小型のマイナスドライバを使ってセットします。
トグルスイッチは普通に指でセットできます。
アドレスバスの状態を示すLED(ADDRESS　BUS)のA15～A0は全消灯します。



1－3.　データのセット

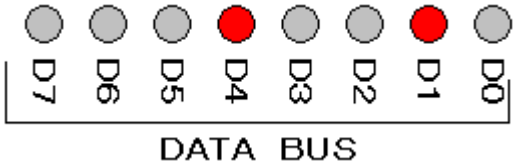
トグルスイッチのD7～D0を使って、データをセットします。
メモリアドレス0000には12（16進数）を書き込みます。
16進数の12は、2進数では0001　0010になります。
図のように、トグルスイッチのD7～D0のうちのD4とD1を上に、その他を下にします。



この段階では、まだデータスイッチの状態はメモリに反映されていません。
メモリからはアドレス0000の現在の値（現在メモリに書き込まれている値）がデータバスに読み出されています。
データバスの状態は、LED（DATA　BUS）のD7～D0に表示されます。
たまたまデータスイッチとメモリからの値とが一致したビットはスイッチの状態がそのままLEDに表示されますが、スイッチとメモリからの値が一致しないビットは、スイッチとメモリからの値がショートするため、LでもなくHでもない中間的な値になりますから、LEDは薄く点灯するかまたは消灯します。
[注意]この状態のまま長く置くことはICにとってはよくありませんから、このまま長い時間放置しないようにしてください。

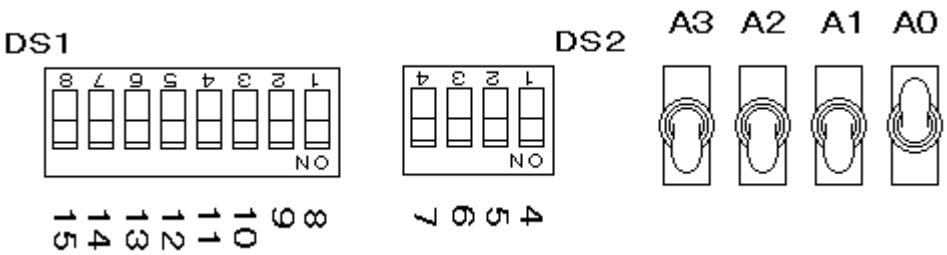
1－4.　MEMWRSWを押す

MEMWRSWを押すと、IC251（74HC123）からワンショットパルスが出力され、そのパルスがメモリ（RAM）のWR信号になるので、データスイッチでセットした値がメモリに書き込まれます。
その瞬間にメモリから出力される値がデータスイッチでセットした値と一致しますから、LED（DATA　BUS）の表示がデータスイッチで設定した値の通りになります。

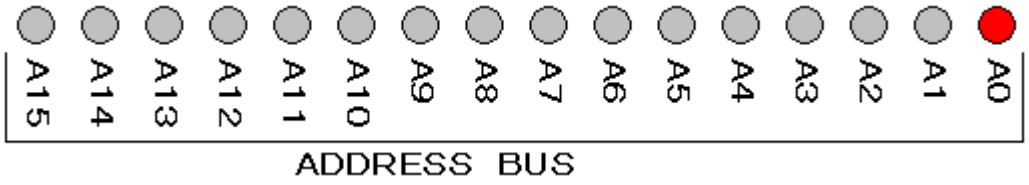


1－5.　次のアドレス0001をセットする

次のアドレス0001（16進数表示）をセットします。
2進数表示では0000　0000　0000　0001ですから、最下位ビットのA0が変わるだけです。
トグルスイッチA0だけを上にセットします。
DS1、DS2は変更なしです。

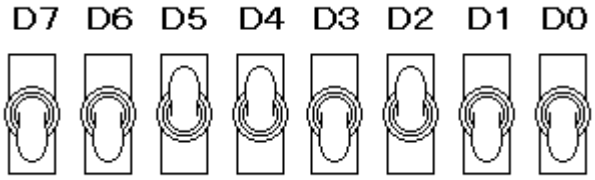


データスイッチD7～D0の値は、MEMWRSWを押さないことには、データバスに反映されませんが、アドレススイッチの状態は、ただちにアドレスバスに反映されます。
アドレスバスの状態を示すLED（ADDRESS　BUS）のA15～A0は、右端のA0のみが点灯します。



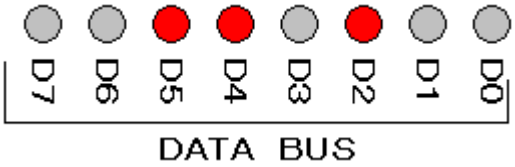
1－6.　データ34をセットする

データ34（16進数）をセットします。
2進数表示では0011　0100です。
トグルスイッチD7～D0のうち、D5、D4、D2が"1"（上側）で、その他のスイッチは"0"（下側）にします。



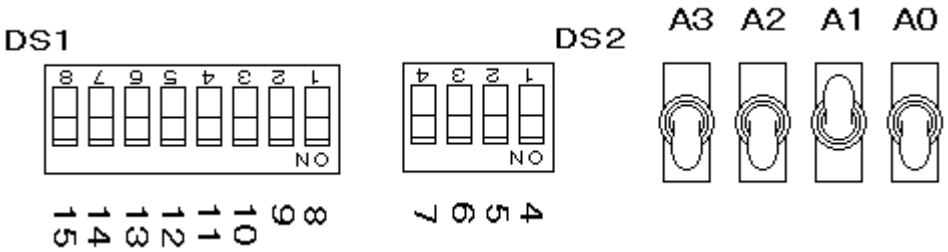
1－7.　MEMWRSWを押す

MEMWRSWを押すとメモリにデータが書き込まれます。
メモリにデータスイッチでセットした値（34）が書き込まれた結果、LED（DATA　BUS）の表示も34（00110100）になります。

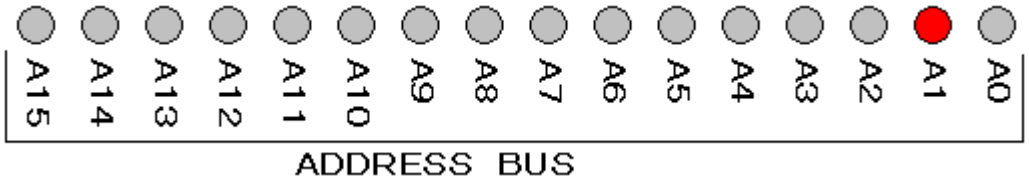


1－8.　次のアドレス0002をセットする

次のアドレス0002（16進数表示）をセットします。
2進数表示では0000　0000　0000　0010ですから、A1のみが"1"（上側）で、その他は"0"（下側）にします。
DS1、DS2は変更なしです。

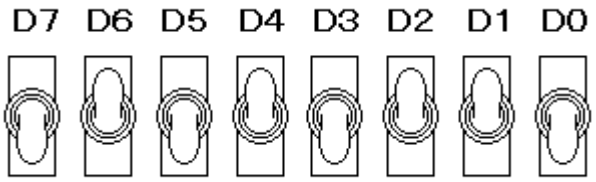


LED（ADDRESS　BUS）の表示も0000　0000　0000　0010になります。



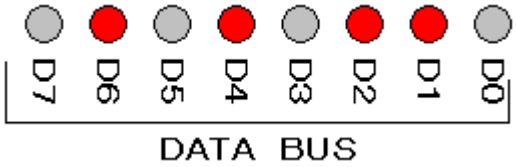
1－9.　データ56をセットする

データ56（16進数）をセットします。
2進数表示では0101　0110です。
トグルスイッチD7～D0のうち、D6、D4、D2、D1が"1"（上側）で、その他のスイッチは"0"（下側）にします。



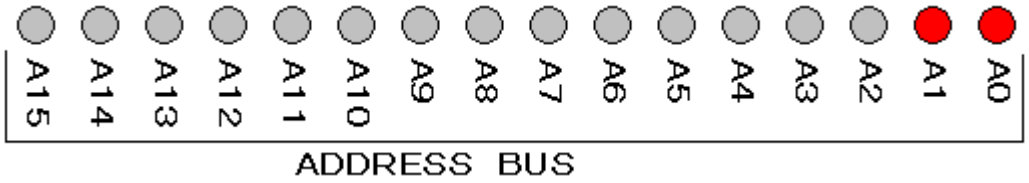
1－10.　MEMWRSWを押す。

　MEMWRSWを押すとメモリにデータが書き込まれます。
　メモリにデータスイッチでセットした値（56）が書き込まれた結果、LED（DATA　BUS）の表示も56（01010110）になります。



1－11.　次のアドレス0003をセットする

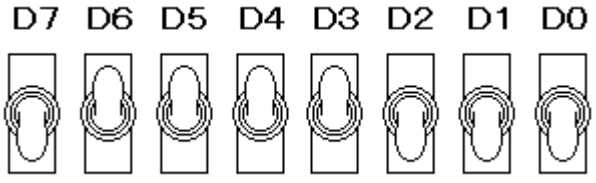
　次のアドレス0003（16進数表示）をセットします。
　2進数表示では0000　0000　0000　0011ですから、A1とA0のみが″1″（上側）で、その他は″0″（下側）にします。
　DS1、DS2は変更なしです。



　LED（ADDRESS　BUS）の表示も0000　0000　0000　0011になります。

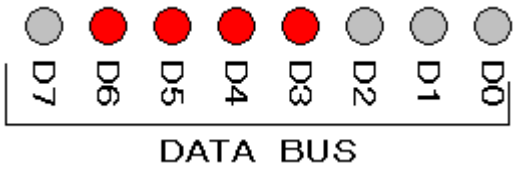
1－12.　データ78をセットする

　データ78（16進数）をセットします。
　2進数表示では0111　1000です。
　トグルスイッチD7～D0のうち、D6、D5、D4、D3が″1″（上側）で、その他のスイッチは″0″（下側）にします。



1－13.　MEMWRSWを押す

　MEMWRSWを押すとメモリにデータが書き込まれます。
　メモリにデータスイッチでセットした値（78）が書き込まれた結果、LED（DATA　BUS）の表示も78（01111000）になります。



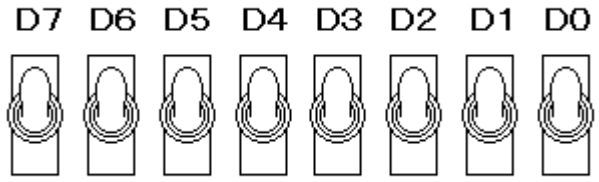
2.　メモリからデータを読む

2－1.　データスイッチをFFにする

ディップスイッチDS3の4をONにした状態では、アドレススイッチ（ディップスイッチDS1、DS2とトグルスイッチA3～A0）で示されるメモリアドレスに書き込まれているデータが、そのままLED（DATA　BUS）のD7～D0に表示されます。

ただしこのときデータスイッチ（トグルスイッチD7～D0）がON（下側）になっていると、そのビットは正しく表示されません。

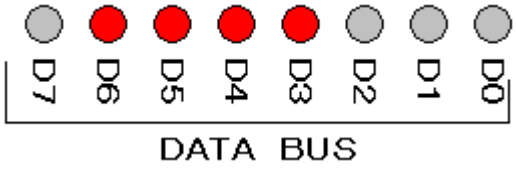
そこでメモリからデータを正しく読むためには、トグルスイッチD7～D0を全て"1"（上側）にしておく必要があります。



データスイッチを図のように全部上側にすると、これは2進数の11111111（16進数のFF）を示していることになります。

データスイッチをこの状態にすると、メモリから出力されたデータがデータスイッチの影響を受けなくなるため、LED（DATA　BUS）に正しく表示されるようになります。

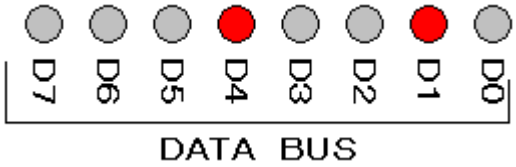
アドレススイッチが、さきほどの1－11. の作業で0003の状態になっていたとすると、データスイッチをFFにしても、LED（DATA　BUS）は78（0111　1000）を表示しているはずです。



2－2.　アドレスを0000にしてみる

アドレススイッチを、1－2. と同じように、0000（16進数）にしてみます。

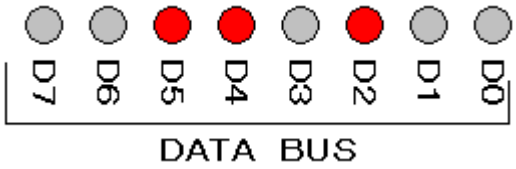
すると、アドレススイッチをそのようにセットするだけで、LED（DATA　BUS）には、さきほどメモリアドレス0000に書き込んだ値12（00010010）がただちに表示されます。



2－3.　アドレスを0001にしてみる

アドレススイッチを、1－5. と同じように、0001（16進数）にしてみます。

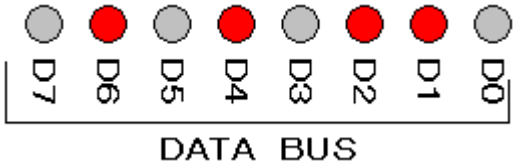
LED（DATA　BUS）には、メモリアドレス0001に書き込んだ値34（00110100）が、ただちに表示されます。



2－4.　アドレスを0002にしてみる

アドレススイッチを、1－8. と同じように、0002（16進数）にしてみます。

LED（DATA　BUS）には、メモリアドレス0002に書き込んだ値56（01010110）が、ただちに表示されます。



3. MVI命令のテストプログラムをメモリに書く

MYCPU80組立説明書　Ⅲ組立　[5]MOV、MVI、HLT命令回路　の組立作業後に行う動作テストの説明です。

3－1.　プログラムの説明

メモリに次のプログラムを書きます。

```
0000  0600    MVI  B, 00
0002  0E01    MVI  C, 01
0004  1602    MVI  D, 02
0006  1E03    MVI  E, 03
0008  2604    MVI  H, 04
000A  2E05    MVI  L, 05
000C  3E07    MVI  A, 07
000E  76       HLT
```

（リスト1）　MVI　r　テストプログラム

上のプログラムリストのうち、本当に必要なマシン語の命令だけを取り出すと、06000E0116021E0326042E053E0776だけになります。

この16進数を、メモリの0000番地から順に書くだけでCPUには十分な命令プログラムになるのですが、でもこれでは人間には全く理解ができません。

そこで少しでも人間にも理解しやすいプログラムになるように、上のリストのような表現にするのが一般的です。

リストの最初の4桁はメモリアドレスです。

8080は0000～FFFFの範囲のメモリアドレスをアクセスしますから、リストのようにメモリアドレスは4桁の16進数で示します。

8080はメモリの0000番地の命令を最初に読み込みますから、普通のマシン語のプログラムは0000番地から書き始めます。

メモリにはマシン語の命令も、 1. メモリにデータを書き込む　で説明したデータの書き込みと同じように1バイトずつ書き込んでいきます。

それならば、

```
0000  06    MVI  B, 00
0001  00
0002  0E    MVI  C, 01
0003  01
   :
   :
```

と書いた方がよいようにも思えますが、マシン語のプログラムの場合には、ニーモニック表現に合わせて命令単位で表記します。

MVI命令は、そのあとに続く1バイトのデータをレジスタ（またはメモリ）に書き込む命令です。

命令の詳しい説明とマシン語コードについては、8080命令説明書を参照してください。

（リスト1）のプログラムをメモリに書き込んで実行させると、Bレジスタに00、Cレジスタに01、Dレジスタに02、Eレジスタに03、Hレジスタに04、Lレジスタに05、Aレジスタに07を書き込んだあとHLT命令を実行し停止します。

「停止する」と書きましたが、厳密に言うと、正確な表現ではありません。

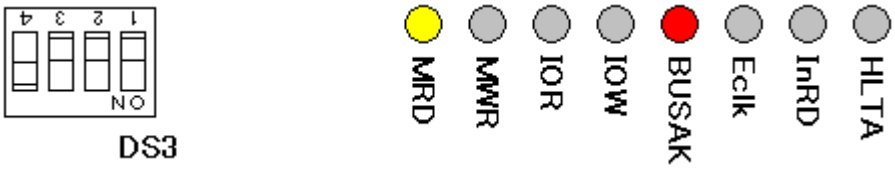
正しくは、そのアドレスから先に進まずに、無限にHLT命令だけを実行し続けますから停止しているように見えます。

この状態を解除するにはリセット信号を入力するしかありません（先にEI命令が実行されていれば、割込みは受け付けられます）。

3－2.　ディップスイッチのセット

プログラムをメモリに書くときは必ずディップスイッチDS3をセットします。
セットの仕方は　1－1. で説明していますが、もう一度説明します。

ディップスイッチDS3の一番左側の4のみを、小型のマイナスドライバを使って図のようにON（下側）にセットします。

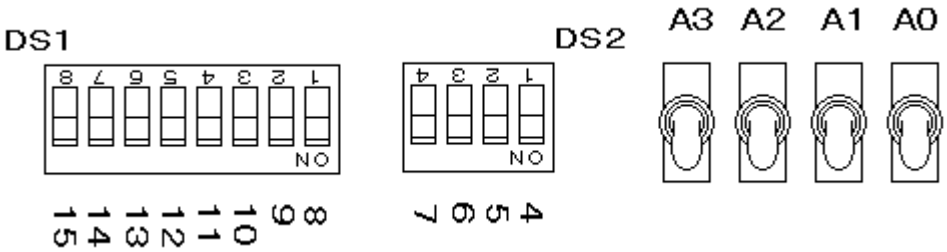


DS3－4をONにすると、アドレスバス、データバスやその他の制御信号ラインがCPUから切り離されて、外部から直接メモリにアクセスすることができるようになります。
BUSが開放されたことを示すLED（BUSAK）が点灯します。同時にメモリがREADアクセス状態になるため、MRD（Memory　Read）も点灯します。

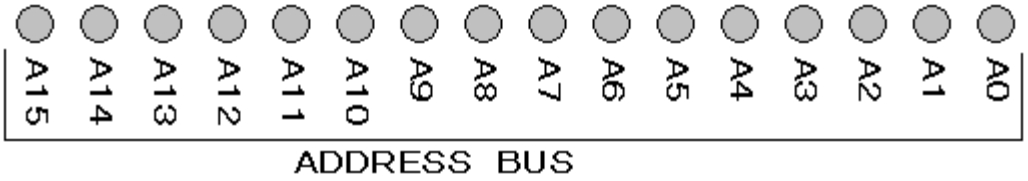
3－3.　アドレスのセット

まず最初のアドレス0000を、ディップスイッチDS1、DS2とアドレス用のトグルスイッチを使ってセットします。
セットの仕方は　1－2.　で説明していますが、もう一度説明します。

図のようにディップスイッチのDS1、DS2とトグルスイッチのA3～A0の全部を下側にします。スイッチを上側にすると"1"、下側にすると"0"がセットされます。



これでアドレスが2進数表示の0000　0000　0000　0000（16進数表示の0000）にセットされたことになります。
ディップスイッチは小さいので、小型のマイナスドライバを使ってセットします。
トグルスイッチは普通に指でセットできます。
アドレスバスの状態を示すLED（ADDRESS　BUS）のA15～A0は全消灯します。

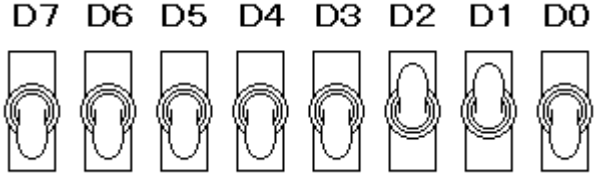


このときデータ入力用のトグルスイッチが全部上側になっておれば、メモリのアドレス0000の値がデータ表示用LED（DATA BUS）に表示されます。

3－4.　命令コードをセットする

アドレス0000に命令コード06（00000110）を書き込むため、データ入力用トグルスイッチをセットします。
1－3. で説明した方法と全く同じです。

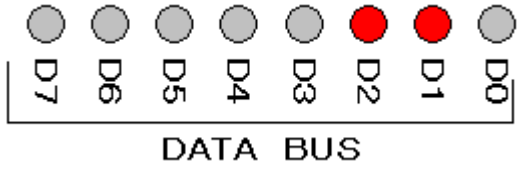
図のように、トグルスイッチのD7～D0のうちのD2とD1を上に、その他を下にします。



この段階では、まだデータスイッチの状態はメモリに反映されていません。
メモリからはアドレス0000の現在の値（現在メモリに書き込まれている値）がデータバスに読み出されています。
データバスの状態は、LED（DATA BUS）のD7～D0に表示されます。
たまたまデータスイッチとメモリからの値とが一致したビットはスイッチの状態がそのままLEDに表示されますが、スイッチとメモリからの値が一致しないビットは、スイッチとメモリからの値がショートするため、LでもなくHでもない中間的な値になりますから、LEDは薄く点灯するかまたは消灯します。

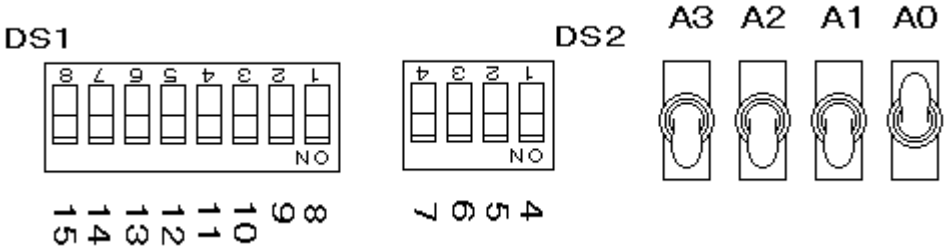
3－5. MEMWRSWを押す

MEMWRSWを押すと、データスイッチでセットした値がメモリに書き込まれます。
その瞬間にメモリから出力される値がデータスイッチでセットした値と一致しますから、LED（DATA BUS）の表示がデータスイッチで設定した値の通りになります。



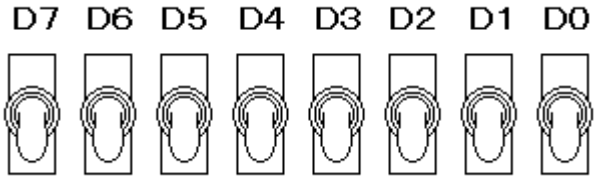
3－6. 次のアドレス（0001）をセットする

3－3. で説明したのと同じ方法で次のアドレスをセットします。
アドレス0001は2進数では 0000 0000 0000 0001 ですから、トグルスイッチのA0だけを上にします。



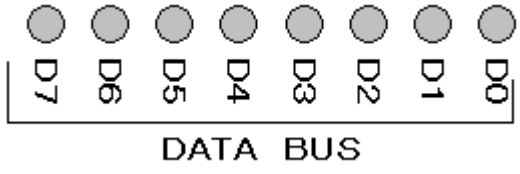
3－7. データの00をセットする

アドレス0001には命令MVI B, 00のデータ部分00を書き込みます。
3－4. で説明したことと同じですが、今度は00（00000000）ですからデータ用のトグルスイッチを全て下にします。



3－8. MEMWRSWを押す

MEMWRSWを押すと、データスイッチでセットした値00がメモリに書き込まれます。
その瞬間にメモリから出力される値がデータスイッチでセットした値と一致しますから、LED（DATA BUS）の表示がデータスイッチで設定した値の通りになります（全部消灯します）。



このあとも同じようにして、アドレス0002～000Eまで命令コード、データを書き込みます。

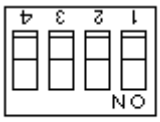
3－9.　メモリに書き込んだプログラムを確認する

メモリにプログラムが正しく書き込まれたかどうか、念の為に確認をしておくとよいでしょう。
確認の仕方は、**2.　メモリからデータを読む**　で説明していますから、そちらを参照してください。

4.　テストプログラムを実行する

4－1.　ディップスイッチ（DS3）のセット

プログラムを実行するときは、ディップスイッチDS3を図のように全部OFF（上側）にセットします。
［注意］必ずRESETSWを押しながらディップスイッチをセットしてください。
メモリにプログラムを書き込んだあとは、いきなりディップスイッチを切り換えると、CPUがただちに実行を開始します。そのときの状態によっては正しく実行されないで、暴走してしまう可能性がありますから、必ずRESETSWを押したまま、DS3を切り換えるようにします。

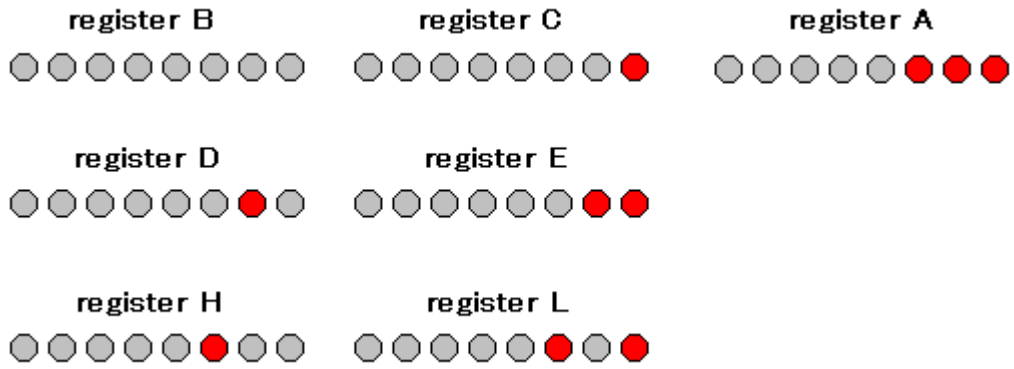


DS3

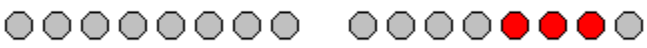
4－2.　リセットスイッチを離す

ディップスイッチDS3を全部OFFにしてから、RESERSWを離します。
するとただちにプログラムが実行され、最後のHLT命令で停止します。
最後にHLT命令を書き忘れると、プログラムが暴走してしまうので、正しい結果が得られません。

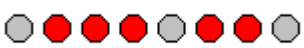
レジスタのLEDは次のように表示されます。
［注］図のサイズと用紙サイズの関係で、実際の基板上のレジスタの配置とは異なっています。



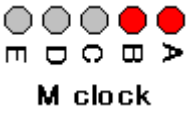
このときPC（プログラムカウンタ）とアドレスバスのLEDは下のように表示されます。



アドレス000EのHLT命令が繰り返し実行されているからです。
データバスLEDとOPコードレジスタの表示は、下のようにHLT命令のコード76（01110110）になっています。



実はこのときCPUは完全に停止しているのではなくて、アドレス000EのHLT命令の、T0～T3のステップを繰り返し実行しています。
その様子は、クロックLEDのE～Aの状態から知ることができます。



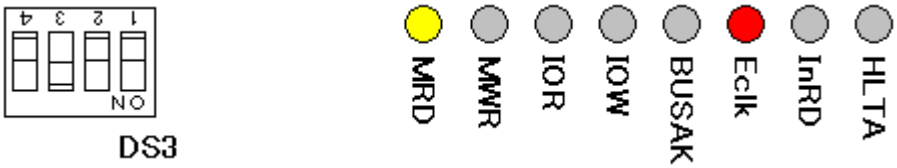
図のように、AとBだけが点灯しているように見えます。
マシンクロックの00000～00011（T0～T3）が高速で実行されているため、このように見えます。

5. ステップ動作

MYCPU80組立説明書　Ⅲ組立　[5]MOV、MVI、HLT命令回路　の組立作業後に行う動作テストの説明の続きですが、同時にステップ動作についての説明も兼ねています。
ここでは、**3. MVI命令のテストプログラムをメモリに書く**　でメモリに書き込んだ、リスト1のプログラムがそのままメモリに残っているものとします。

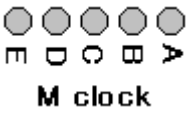
5－1.　ディップスイッチ（DS3）のセット

ステップ動作をさせるときは、ディップスイッチDS3の右から3番目の3のみを、小型のマイナスドライバを使って図のようにON（下側）にセットします。
[注意]必ずRESETSWを押しながらディップスイッチをセットしてください。



DS3－3をONにすると、LED（Exclk／STEP）が点灯します。
CPUのクロックが水晶発振回路から切り離されて、クロックのソースが外部からの入力パルスになったことを示しています（ステップモード）。

この状態で、RESETSWを離してもCPUはアドレス0000で停止しています。
PC（プログラムカウンタ）も外部アドレスバスも、アドレス0000を表示していることを確認してください。
MclockのA～EのLEDも全て消灯していて、マシンクロックがT0の状態になっています。



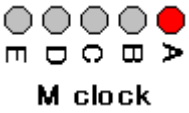
T0では、PC（プログラムカウンタ）の値が外部アドレスバスに出力され、メモリからそのアドレスの値が読み出されます（PCADoutが点灯して、プログラムカウンの値が外部アドレスバスに出力されていることを示します）。
DATA　BUS（外部データバス）には、メモリアドレス0000から読み出された値が表示されます。
3. で入力したプログラムのままならば、アドレス0000にはMVI　B命令のコード06が書かれていますから、DATA BUSには06（00000110）が表示されています。

その値はinner　BUS（内部データバス）に読み込まれます。外部データバスと内部データバスの間にあるゲート（IC231、74HC245）の状態を示すLED、DataGとDataDIRが点灯して、データバスケートがアクティブ（DataGがオン）で、その向きが外部→内部（DataDIRがオン）であることを示します。

T0では、データバスの値はまだOPcode　register（命令コードレジスタ）にはラッチされません。
T0は命令の最初のステップですが、準備のためのステップで命令回路には影響を与えません。

5－2.　STEPSW

ステップモードのときにSTEPSWを押すと、押すたびにCPUクロック回路に1クロックパルスが与えられ、マシンクロックのタイミングが1ステップ進みます。
STEPSWを押す前がT0の状態だったとすると、STEWPSWを押すとT1に進みます。



T1のときに、DATA BUS（データバス）の値（命令コード）がOPcode register（命令コードレジスタ）に読み込まれます。

OPftch（OPcode fetch）が点灯して、命令コードレジスタに値をラッチするための信号がアクティブであることを示します（まだラッチはされません。値は筒抜けになっています）。

このとき、命令コードレジスタに読み込まれた値が命令回路に伝わります。

この説明の例では、メモリアドレス0000に書かれた命令MVI Bの命令コード06（00000110）が、命令回路に伝わります。

命令コードはビットに分解され、ソースレジスタ（データを読み出すレジスタ）を示すs3～s0の信号と、デスティネーションレジスタ（データを書き込むレジスタ）を示すd3～d0信号が作り出されます。

s3～s0、d3～d0とレジスタの関係を以下に示します。

（表1） s3～s0、d3～d0とレジスタの関係

	s3-s0,d3-d0			
register	3*	2	1	0
A	1	1	1	1
F（M）*	1	1	1	0
L	1	1	0	1
H	1	1	0	0
E	1	0	1	1
D	1	0	1	0
C	1	0	0	1
B	1	0	0	0
WKL	0	1	1	1
WKH	0	1	1	0
SPL	0	1	0	1
SPH	0	1	0	0
PCL	0	0	1	1
PCH	0	0	1	0

*1110はF（フラグレジスタ）またはM（メモリ）をともに示します
*ビット3は汎用8ビットレジスタと特殊レジスタを区別するために付加されます

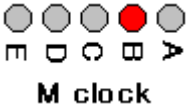
8ビットの命令コードのビットのうち、下位3ビット（bit2～bit0）はs2～s0を示し、その上位の3ビット（bit5～bit3）はd2～d0を示します。

命令コード06（00000110）のとき、d2～d0は000、s2～s0は110になります。

d3、s3はT4以降のステップで回路によって確定されます。T1～T3の期間中は、ともに1になります。

T1のとき、d3～d0、s3～s0の状態を示すLEDが下図のように表示されることを確かめてください。

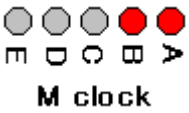
ここでもう一度STEPSWを押すと、マシンクロックが1クロック進んでT2になります。



T2は命令コードレジスタに命令コードをラッチ確定するためのステップです。

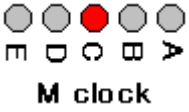
OPftchが消灯して、命令コードのフェッチサイクルが完了したことを示します（OPftchがONの期間中はOPコードレジスタはまだ値をラッチしてはいなくて、値は筒抜け状態になっています）。

もう一度STEPSWを押すと、マシンクロックが1クロック進んでT3になります。

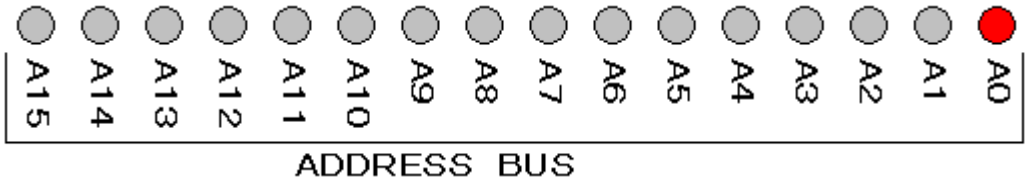


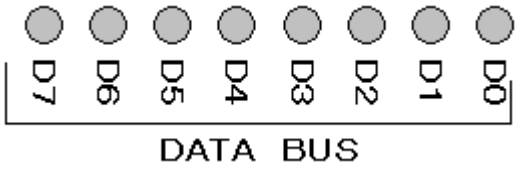
T3のときPC（プログラムカウンタ）をインクリメント（＋1）するためのパルスがONになります。
PCclkが点灯してプログラムカウンタのクロック入力が↓になったことを示します。
この段階ではまだプログラムカウンタはインクリメントしません。
T3が終了したとき（次のT4になった瞬間）に、インクリメントされます。
以上のT0～T3は特殊な命令を除いたほとんど全ての命令で動作は同じです。
つまりこの期間はまだ具体的な命令の解読動作は行われません。

さらにSTEPSWを押してT4に進むと、命令によって個別の状態になります。



PCclkが消灯して、PC（プログラムカウンタ）が0001になることを確認してください。
同時に外部アドレスバスも0001になり、外部データバスにはメモリアドレス0001のメモリの値が出力されます。





以後の動作については説明を省略しますが、MVI　B命令では、このあとT5まで進んだところで命令の動作が終了し、次のT6は次の命令のT0になります。

以上がステップ動作の基本的な操作です。

6. MVI命令のテスト（2）

MYCPU80組立説明書　Ⅲ組立　[5]MOV、MVI、HLT命令回路　の組立作業後に行う動作テストの説明です。

6－1.　MVI　M命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  2604    MVI  H, 04
0002  2E05    MVI  L, 05
0004  36AB    MVI  M, AB
0006  76      HLT
```

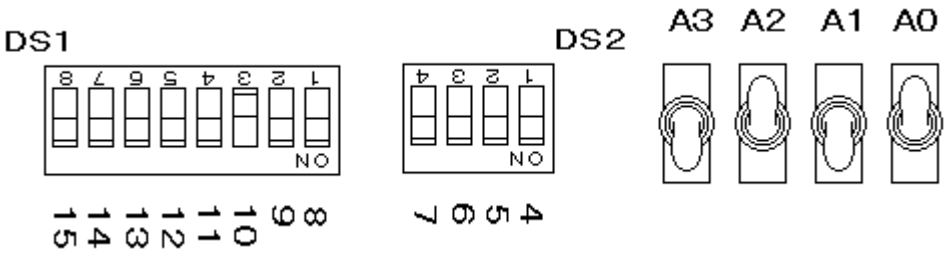
（リスト2） MVI M テストプログラム

MVI M命令は、命令コード36の次のアドレスに書かれている値を、HLレジスタで示すメモリアドレスに書き込みます。
リスト2ではHレジスタに04、Lレジスタに05を入れたあとで、MVI M, ABを実行しますから、メモリアドレス0405にABが書き込まれることになります。

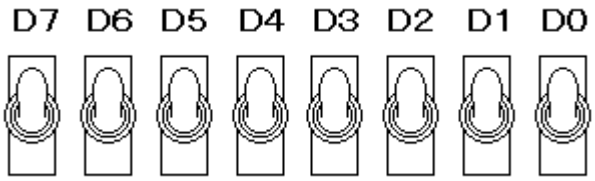
6－2. メモリアドレス0405の値を確認する

プログラムの実行前と実行後でメモリアドレス0405の内容が書き換わることを確認するため、プログラムの実行前のメモリアドレス0405の値を確認しておきます。

ディップスイッチDS1、DS2とアドレス設定用のトグルスイッチを下のようにセットします。



データ入力用のトグルスイッチは全部上側にします（そのようにしないと、メモリから出力されたデータを正しく確認することができません）。



するとDATA BUS（外部データバス）LEDには、メモリアドレス0405の値が表示されます。このときに表示される値は現在アドレス0405に入っている値ですが、どのような値が入っているかはわかりません。
テストを行う上では、AB以外の値であれば、どのような値になっていても構いません。
もしもアドレス0405の値がたまたまABになっていたら、適当な値に書き換えたあと、またデータ設定用トグルスイッチをFFにセットしておいてください。

6－3. テストプログラムを実行する

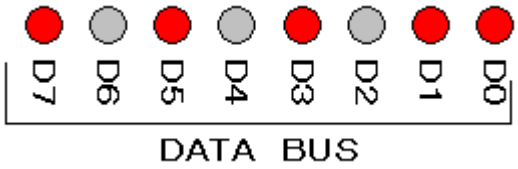
以上の準備ができたら、6－1. でメモリに書いたテストプログラムを実行します。
プログラムを実行したあとで、またメモリの値を確認しますから、アドレス設定用のディップスイッチやトグルスイッチは0405をセットしたままにしておいてください（データ設定用のトグルスイッチはFFにしておきます）。

テストプログラムの実行の仕方については **4. テストプログラムを実行する** を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。
DS3－4がOFFのときはアドレス設定用のDS1、DS2、トグルスイッチやデータ設定用のトグルスイッチはONになっていても、プログラムの実行には影響を与えません。
プログラムは瞬時に最後まで実行されます。
プログラムの最後はアドレス0006のHLT命令（コード76）になっているため、プログラムカウンタと外部アドレスバスには0006が表示され、外部データバス、内部データバス、命令コードレジスタには76が表示されます。
M clockは、AとBが点灯しているように見えます。アドレス0006のHLT命令がT0～T3のステップを繰り返し実行されていることを示しています。

6－4.　メモリアドレス0405の値を確認する

プログラムが正しく実行されたことを確認するために、もう一度メモリアドレス0405の値を確認してみます。
DS3－4をONにします（RESETSWを押しながら、DS3をセットしてください）。

プログラムが正しく実行されていれば、DATA　BUS（外部データバス）LEDには、書き換えられたメモリアドレス0405の値ABが表示されます（そのように表示されなかったら、アドレス設定用のディップスイッチ、トグルスイッチが0405になっていて、データ設定用のトグルスイッチがFFになっているかどうかを確認してください）。



7. MOV命令のテスト

MYCPU80組立説明書　Ⅲ組立　[5]MOV、MVI、HLT命令回路　の組立作業後に行う動作テストの説明です。

7－1.　MOV命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。
6. のテストを行った後で、このテストをする場合には、アドレス0005までは6. のリスト2と同じですから、0006から書くようにしても構いません。

```
0000  2604    MVI  H, 04
0002  2E05    MVI  L, 05
0004  36AB    MVI  M, AB
0006  46      MOV  B, M
0007  75      MOV  M, L
0008  4C      MOV  C, H
0009  76      HLT
```

（リスト3）　MOV　テストプログラム

7－2.　テストプログラムを実行する

テストプログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
プログラムは瞬時に最後まで実行されます。
プログラムの最後はアドレス0009のHLT命令（コード76）になっているため、プログラムカウンタと外部アドレスバスには0009が表示され、外部データバス、内部データバス、命令コードレジスタには76が表示されます。
M　clockは、AとBが点灯しているように見えます。アドレス0006のHLT命令がT0～T3のステップを繰り返し実行されていることを示しています。

プログラムを実行すると、メモリアドレス0405（M）にデータABが書き込まれた後、同じメモリアドレスから読み出された値（AB）が、レジスタBに入れられます。
その後メモリアドレス0405には、レジスタLの値（05）が入れられます。
最後にレジスタCにレジスタHの値（04）が入れられたあとHLT命令が実行されます。
テストプログラムの実行後は、レジスタBのLED表示はABになり、レジスタCのLED表示は04になります。

7－3.　メモリアドレス0405の値を確認する

6－4. と同じように操作して、メモリアドレス0405の値を確認してみてください。
レジスタLの値、05が書き込まれていることが確認できます。

8. INR命令のテスト

MYCPU80組立説明書　Ⅲ組立　[6]INR／DCR命令回路　の組立作業後に行う動作テストの説明です。

8－1．　最初に電源を一度OFFにする

今回のテストはレジスタを強制的にクリアしてから始めたいので、最初にこういうことをします。
メモリ書き込みモードの状態で（DS3－4をONにしたままで）、一旦電源をOFFにします。
そうすることによって、レジスタの値をFFにするためです。

電源を一度切って、数秒待ってから再投入します。
すると、プログラムは電池によってバックアップされていますから消えませんが、レジスタの値は消えてしまいます。
電源がONになったとき、レジスタの値はたいていはFFになります（必ずFFになるとは限りません）。
レジスタの状態をクリアしておいてから、メモリにテストプログラムを書きます。

[注記]ここでは先に電源がONになっていて、何かのプログラムがすでに実行されていて、レジスタの値がレジスタごとにばらばらになっていることを想定して、この作業をするように書きましたが、現在電源がOFFの状態で、これからプログラムの書き込みテスト作業をするために電源をONにする場合には、あらためて電源のOFF／ONを繰り返す必要はありません。

8－2．　INR命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  04       INR  B
0001  0C       INR  C
0002  14       INR  D
0003  1C       INR  E
0004  24       INR  H
0005  2C       INR  L
0006  3C       INR  A
0007  76       HLT
```

（リスト4） INR r テストプログラム

8－3．　テストプログラムを実行する

テストプログラムの実行の仕方については　**4．　テストプログラムを実行する**　を参照してください。

プログラムが実行されると、B、C、D、E、H、L、Aの各レジスタが＋1されます。
プログラム実行前のレジスタの値がFFだったとすると、FF＋1＝00なので、プログラムの実行後は00が表示されます。その状態でプログラムは停止して、HLT命令を実行し続けます。

8－4．RESETSWを押す

RESETSWを押す（押してから離す）と、再びプログラムが実行され、レジスタの値が＋1されて表示されます。
RESETSWを押すたびに（押してから離すごとに）、プログラムが実行されて、レジスタが＋1されます。

プログラムの先頭で、レジスタにMVI命令で初期値を与えるようにはしないで、電源OFFによってレジスタをクリアしたのは、こういう動作をさせたかったからです。

9．DCR命令のテスト

MYCPU80組立説明書　Ⅲ組立　[6]INR／DCR命令回路　の組立作業後に行う動作テストの説明です。

9－1．　DCR命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  05       DCR  B
0001  0D       DCR  C
0002  15       DCR  D
```

```
0003  1D      DCR  E
0004  25      DCR  H
0005  2D      DCR  L
0006  3D      DCR  A
0007  76      HLT
```
（リスト5） DCR r テストプログラム

9－2. テストプログラムを実行する

テストプログラムの実行の仕方については **4. テストプログラムを実行する** を参照してください。

さきほどのプログラム（リスト4）はINR命令のプログラムでしたが、今度はDCR命令のプログラムです。
プログラムが実行されると、B、C、D、E、H、L、Aの各レジスタが－1されたあと、その状態で停止します（HLT命令を実行し続けます）。

8－4. と同じように、RESETSWを押すと、今度は押すたびに（押して離すたびに）、レジスタが－1されます。
プログラム実行前のレジスタの値が00だったとすると、00－1＝FFなので、プログラムの実行後はFFが表示されます。

10. INR M命令のテスト

MYCPU80組立説明書 Ⅲ組立 [6]INR／DCR命令回路 の組立作業後に行う動作テストの説明です。

10－1. INR M命令のテストプログラムをメモリに書く

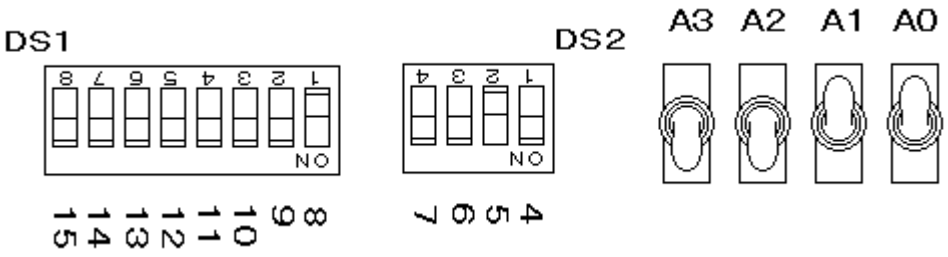
メモリに次のプログラムを書きます（**1. メモリにデータを書き込む** の説明を参考にして操作してください）。

```
0000  2601    MVI  H, 01
0002  2E23    MVI  L, 23
0004  34      INR  M
0005  46      MOV  B, M
0006  76      HLT
```
（リスト6） INR M テストプログラム

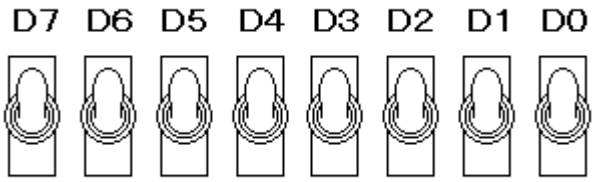
10－2. メモリアドレス0123の値を確認する

今度のプログラムはメモリの値をインクリメント（＋1）します。
プログラムの実行前と実行後でメモリアドレス0123の内容が書き換わることを確認するために、プログラムを実行する前に、メモリの値を見ておきます。

ディップスイッチDS1、DS2とアドレス設定用のトグルスイッチを下のようにセットします。



データ入力用のトグルスイッチは全部上側にします（そのようにしないと、メモリから出力されたデータを正しく確認することができません）。



このようにすると、DATA　BUS（外部データバス）LEDには、メモリアドレス0123の値が表示されます。
このときに表示される値は現在アドレス0123に入っている値ですが、どのような値が入っているかはわかりません。
テストを行う上では、どのような値になっていても構いません。

10－3.　テストプログラムを実行する

メモリアドレス0123の値を確認したあと、10－1. でメモリに書き込んだプログラムを実行します。
プログラムを実行したあとで、またメモリの値を確認しますから、アドレス設定用のディップスイッチやトグルスイッチは0123をセットしたままにしておいてください（データ設定用のトグルスイッチはFFにしておきます）。

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。
DS3－4がOFFのときはアドレス設定用のDS1、DS2、トグルスイッチやデータ設定用のトグルスイッチはONになっていても、プログラムの実行には影響を与えません。

テストプログラム（リスト6）は、HLで示されるメモリアドレス0123の値を＋1したあと、その値をレジスタBに入れます。プログラムは瞬時に実行され、停止します（最後のHLT命令が繰り返し実行されます）。
この状態のままではメモリアドレス0123の値を直接見ることはできませんが、レジスタBのLED表示を見ることでメモリの値がインクリメントされたことを確認することができます。

10－4. RESETSWを押す

RESETSWを押す（押してから離す）と、再びプログラムが実行され、レジスタBの値が＋1されて表示されます。
RESETSWを押すたびに（押してから離すごとに）、プログラムが実行されて、レジスタBが＋1されます。
このことでメモリアドレス0123の値がプログラムの実行によって＋1されていく様子を知ることができます。

10－5.　メモリアドレス0123の値を確認する

レジスタBのLED表示によって、メモリアドレス0123の値がプログラムの実行のたびにインクリメントされることを間接的に知ることはできるのですが、念の為にメモリアドレス0123の値を直接確認してみることにします。
DS3－4をONにします（RESETSWを押しながら、DS3をセットしてください）。

アドレス設定用のディップスイッチ、トグルスイッチが0123になっていて、データ設定用のトグルスイッチがFFになっていれば、メモリアドレス0123の値がDATA　BUS（外部データバス）LEDに表示されます。

11. DCR　M命令のテスト

MYCPU80組立説明書　Ⅲ組立　[6]INR／DCR命令回路　の組立作業後に行う動作テストの説明です。

11－1.　DCR　M命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  2601    MVI  H, 01
0002  2E23    MVI  L, 23
0004  35      DCR  M
0005  46      MOV  B, M
0006  76      HLT
```

（リスト7）　DCR　M　テストプログラム

10．で書いたINR　Mテストプログラム（リスト6）の0004が34から35に変わるだけですから、もし10．の作業をした直後ならば、メモリアドレス0004に35を書き込むだけで構いません。

11－2．　テストプログラムを実行する

　プログラムの実行の仕方については　**4．　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　テストプログラム（リスト7）は、HLで示されるメモリアドレス0123の値を－1したあと、その値をレジスタBに入れます。プログラムは瞬時に実行され、停止します（最後のHLT命令が繰り返し実行されます）。
　この状態のままではメモリアドレス0123の値を直接見ることはできませんが、レジスタBのLED表示を見ることでメモリの値がデクリメントされたことを確認することができます。

11－3．RESETSWを押す

　RESETSWを押す（押してから離す）と、再びプログラムが実行され、レジスタBの値が－1されて表示されます。
　RESETSWを押すたびに（押してから離すごとに）、プログラムが実行されて、レジスタBが－1されます。
　このことでメモリアドレス0123の値がプログラムの実行によって－1されていく様子を知ることができます。

12．JMP命令のテスト

　MYCPU80組立説明書　Ⅲ組立　［7］JMP命令回路　の組立作業後に行う動作テストの説明です。

12－1．　JMP命令のテストプログラムをメモリに書く

　メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  3E00     MVI  A, 00
0002  47       MOV  B, A
0003  4F       MOV  C, A
0004  57       MOV  D, A
0005  5F       MOV  E, A
0006  0C       INR  C
0007  C20600   JNZ  ＄0006
000A  04       INR  B
000B  C20600   JNZ  ＄0006
000E  1C       INR  E
000F  C20600   JNZ  ＄0006
0012  14       INR  D
0013  C30600   JMP  ＄0006
```

（リスト8）　JMP　テストプログラム

［注記］JNZとJMP命令のジャンプ先アドレスが0006ではなくて＄0006になっていますが、これは当社オリジナルの8080アセンブラで16進数アドレスを表記するときのルールです。

12－2．　テストプログラムを実行する

　プログラムの実行の仕方については　**4．　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　このプログラムは8ビットのレジスタC、B、E、Dを直列につないで32ビットのカウンタにしたものです。最初にCレジスタが00からFFまでインクリメントし、FFから00になるときに、その上位にあるBレジスタがインクリメントされます。そのようにして、C、B、E、Dの順にカウントが進みます。

カウントが進む様子はレジスタのLEDで目視することができますが、CレジスタとBレジスタは非常に高速なのでほとんど全点灯しているようにしか見えません。
Cレジスタが＋1カウントアップするのにかかる時間は、INR　C命令とJNZ　＄0006命令の実行時間を加算したものになります。

8080命令説明書から、INR　Cの実行クロック数は8クロック、JNZ命令の実行クロック数は12クロックまたは8クロックです。JNZ命令はCが00になるときだけは8クロックですがその他の場合には12クロックです。
MYCPU80のCPUクロックは2MHzですから、1マシンクロックは0．5μSです。
ですからCレジスタが＋1カウントアップするのにかかる時間は、
（8＋12）×0．5＝10μSということになります。

Cレジスタが00から次の00までカウントアップして、その上位のBレジスタが＋1カウントアップするのにかかる時間は、その257倍（Bレジスタの＋1アップの分も含める）ですから、2570μS（2．57mS）になります。
JNZ命令は256回に1回は8クロックになりますから、正確に計算すると、2570－4×0．5＝2568μSです。

Bレジスタが00から次の00までカウントアップして、その上位のEレジスタが＋1カウントアップするのにかかる時間は、2．568×256≒657．4mSになります。

[注記]上記の計算の詳細については、当社ホームページの
「つくるCPU[第148回]」（http://www.alles.or.jp/~thisida/mycpu148.html）
を参照してください。

Eレジスタは657．4mSに1回カウントアップされますから、LEDに表示されるEレジスタの値に657．4mSを掛けることで、プログラム開始からの経過時間を求めることができます。

たとえば、Eレジスタが00から次の00までカウントアップして、その上位のDレジスタが＋1されたとき、つまりDレジスタが1でEレジスタが00になったときのスタートからの経過時間は、次の計算によって求められます。
16進数の100は10進数の256ですから、
657．4×256＝168294．4mS
約168．3秒になります。

テストプログラム（リスト8）はJMP命令とJNZ命令をテストするだけのプログラムです。
条件JMP命令にはJNZのほかにもJZ、JNC、JC、JPO、JPE、JP、JMの各命令があります。
しかしそれらの命令はデコード回路以外ではJNZと同じ回路になっていますから、JNZが正しく動作すればそのほかの条件JMP命令も正しく動作すると考えてもよいでしょう。
そこでここではJNZ以外の条件JMP命令のテストは省略することにしました（全ての命令をテストするテストプログラムは、TK80回路完成後に実行できるものを用意してあります）。

13．LXI命令のテスト

MYCPU80組立説明書　Ⅲ組立　[8]LXI、PUSH、POP、CALL、RET命令回路　の組立作業後に行う動作テストの説明です。

13－1．　LXI命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000　013412　　LXI　B，＄1234
0003　117856　　LXI　D，＄5678
0006　21BC9A　　LXI　H，＄9ABC
0009　31F0DE　　LXI　SP，＄DEF0
000C　76　　HLT
```
（リスト9）　LXI　テストプログラム

プログラムは非常に簡単なものです。
ペアレジスタBC、DE、HLとSP（スタックポインタ）に2バイトの値を入れるだけのプログラムです。

13－2.　テストプログラムを実行する

　プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　プログラムが実行されると、BCレジスタに1234が、DEレジスタに5678が、HLレジスタに9ABCが、SP（スタックポインタ）にDEF0がそれぞれ入れられます。
　その状態でプログラムは停止して、HLT命令を実行し続けます。

14. LXI、PUSH、POP命令のテスト

　MYCPU80組立説明書　Ⅲ組立　[8]LXI、PUSH、POP、CALL、RET命令回路　の組立作業後に行う動作テストの説明です。

14－1.　LXI、PUSH、POP命令のテストプログラムをメモリに書く

　メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000     LXI  SP, $0000
0003  3E55       MVI  A, 55
0005  013412     LXI  B, $1234
0008  117856     LXI  D, $5678
000B  21BC9A     LXI  H, $9ABC
000E  C5         PUSH  B
000F  D5         PUSH  D
0010  E5         PUSH  H
0011  F5         PUSH  PSW
0012  F1         POP  PSW
0013  E1         POP  H
0014  D1         POP  D
0015  C1         POP  B
0016  C30E00     JMP  $000E
```

（リスト10）　LXI、PUSH、POP　テストプログラム

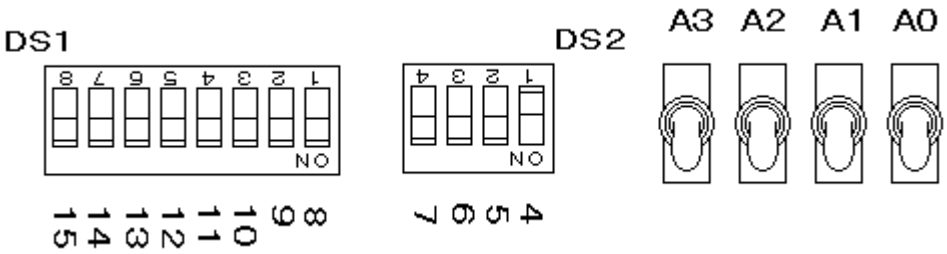
　このプログラムはリスト9にPUSH命令とPOP命令を追加したものです。
　PUSH命令とPOP命令を繰り返し実行します。

[注意]PUSH命令、POP命令を使うプログラムは、SP（スタックポインタ）の設定が必要です。
　このテストプログラムのようにSP＝0000に設定した場合、スタックはそれ（0000）から－1したFFFFから使われていきます。

[参考]PC（プログラムカウンタ）以外のレジスタはリセットによってクリアされません。

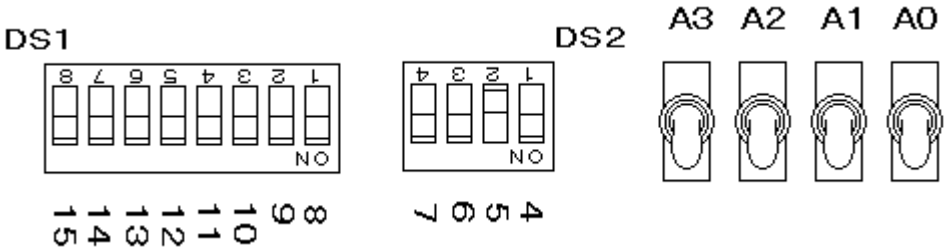
[アドレススイッチの設定]
　今までのプログラムは短いものばかりでしたが、今回は少し長いプログラムです。
　今までのプログラムはアドレスが000Fまでにおさまっていましたから、アドレス設定用のデイップスイッチは全ビットが0（全部下側）でしたが、今回のようにアドレスが000Fを越えるときは、アドレス設定用のディップスイッチも使うことになります。

例）アドレスを0010にする（DS2－1のみを1（上側）にする）。



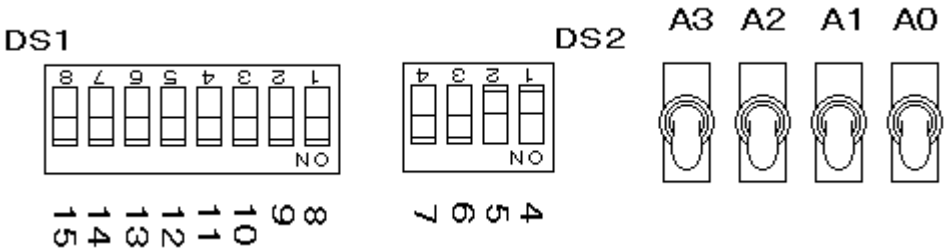
この状態で、トグルスイッチのA3～A0をセットすることによって、アドレス0010～001Fを指定します。

アドレス0020は下の図のように、DS2－2を上に（DS2－1は下に）します。



この状態で、トグルスイッチのA3～A0をセットすることによって、アドレス0020～002Fを指定します。

アドレス0030は下の図のように、DS2－2とDS－1を上にします。



トグルスイッチのA3～A0がアドレスの下位4ビットを設定するために使われているのと全く同じように、ディップスイッチDS2はアドレスのA7～A4の設定に使われ、またディップスイッチDS3はアドレスのA15～A8の設定に使われています。
長いプログラムを0000から順にメモリに書き込んでいくような場合、うっかりして、上位アドレスを設定するDS2、DS1を変更することを忘れてしまうと、プログラムが正しくメモリに書き込まれません。
長いプログラムを書き込むときはこのことによく注意してください。

14－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、BCレジスタに1234が、DEレジスタに5678が、HLレジスタに9ABCが入れられ、Aレジスタには55が入れられます。
その後はレジスタBC、DE、HL、A（とフラグレジスタ）に対するPUSH命令とPOP命令を繰り返し実行し続けます。

レジスタの値はPUSH命令によってスタック（メモリ）に一旦格納されたあと、POP命令によって再びメモリから読み出されてもとのレジスタに書き込まれます。
メモリ、レジスタへの書き込みが繰り返されますが、PUSH命令、POP命令とJMP命令が正しく実行されている限り、レジスタの値は変化しません。

15. CALL、RET命令のテスト

MYCPU80組立説明書　Ⅲ組立　[8]LXI、PUSH、POP、CALL、RET命令回路　の組立作業後に行う動作テストの説明です。

15－1.　CALL、RET命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000     LXI  SP, $0000
0003  3E00       MVI  A, 00
0005  210000     LXI  H, $0000
0008  110000     LXI  D, $0000
000B  3C         INR  A       (8)
000C  CC2000     CZ   $0020   (8  or  18)
000F  C30B00     JMP  $000B   (12)

0020  2C         INR  L     (8)
0021  C0         RNZ      (4  or  8)
0022  24         INR  H     (8)
0023  C0         RNZ      (4  or  8)
0024  1C         INR  E     (8)
0025  C0         RNZ      (4  or  8)
0026  14         INR  D     (8)
0027  C9         RET      (8)
```

（リスト11）　CALL、RET　テストプログラム

このプログラムはCALL命令とRET命令をテストするものです。
CALL命令は別のアドレスにあるサブルーチンをCALLして実行する命令です。
指定するアドレスに移ってそのアドレスから命令を実行する、という動作はJMP命令と同じですが、JMP命令は「行きっぱなし」であるのに対し、CALL命令はRET命令によって、もとのプログラムに戻ってくることができます（JMP命令はRET命令によっても、もとに戻ってくることはできません）。

[注意]CALL命令、RET命令を使うプログラムは、SP（スタックポインタ）の設定が必要です。
このテストプログラムのようにSP＝0000に設定した場合、スタックはそれ（0000）から－1したFFFFから使われていきます。

[注意]サブルーチンプログラムはメモリアドレス0020から書き込みます（ディップスイッチDS2の設定を変えることを忘れないようにしてください）。
サブルーチンをメインプログラムに続けて書いても構わないのですが、マシン語のプログラムの場合には、一度書いてしまうと、後から途中に命令を追加することができません。
途中で命令コードを追加するときは、そこから後ろを全部書き直さなければなりません。
サブルーチンを少し後ろから書くようにすると、プログラム変更の場合の負担が少なくなります。またサブルーチンを「きり」のいいアドレスから始めることで、プログラムが後から見やすいものになります。
なお、このような配慮はマシン語プログラムについてのもので、アセンブラを使ってプログラムを作成する場合にはそれほど意識する必要はありません。

15－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、Aレジスタが00～FFまでインクリメント（＋1）されます。結果が00でないときは、CZ命令は実行されませんから、次のJMP命令によって、また000Bに戻ってINR　Aが実行され、それを繰り返します。
結果が00のときは、0020のサブルーチンがCALLされます。
CZはZフラグがONのときにサブルーチンをCALLする命令です。

0020のサブルーチンではINR　Lによって、Lレジスタが00～FFまでインクリメント（＋1）されます。結果が00でないときは、RNZが実行され、そのままメインルーチンに戻ります。
RNZはZフラグがOFFのときにメインプログラムにリターンする命令です。
結果が00のときは、次のINR　Hが実行され、Hレジスタがインクリメント（＋1）されます。その結果が00でないときは、そこでRNZが実行され、メインルーチンに戻ります。

結果が00のときは、さらにその次のINR Eが実行されます。

そのようにして、順次、A、L、H、E、Dレジスタがインクリメントされていきます。
Aが最下位8ビットでDが最上位8ビットになるように、8ビツトのバイナリカウンタを5個、直列につないだのと同じ動作となります。

テストプログラム(リスト11)の命令の後ろに( )で示した数字は、その命令の実行クロックです。
INR A、CZ、JMPのループの実行クロック数は、INR Aの結果が00ではないとき(つまり結果が01～FFになる255回は)8＋8＋12＝28クロックで、結果が00になるときだけ、8＋18＋12＝38クロックになります。
このことからAレジスタが00から始まって次の00まで256回インクリメントされるときのクロック数は
28×255＋38＝7178クロックになります。
また256回のうち1回だけは10クロック多くかかる、と考えて
28×256＋10＝7178クロックという計算をすることもできます。

MYCPU80のCPUクロックは2MHzですから、1マシンクロックは0．5μSです。
7178クロックを時間に直すと、7178×0．5＝3589μSになります。
Aレジスタが00から次の00までカウントアップされるのに3．6mSしかかかりませんからLEDは全点灯しているようにしか見えません。

その上のレジスタについても計算してみましょう。
Aレジスタが00になるときにCALLされるアドレス0020のサブルーチンでは、Lレジスタがインクリメントされます。
この部分の実行クロック数の計算もAレジスタのときと同じように考えますが、Lレジスタが＋1されるときは、Aレジスタが00～00まで一巡したときであることも計算に含める必要があります。
この計算は256回に1回RNZが実行されないときだけ4クロック少ない、と考えると簡単になります。
(7178＋8＋8)×256－4＝1841660クロックになります。
実行時間は1841660×0．5＝920830μSです。
Lレジスタが00から次の00までカウントアップされるのにかかる時間は920．8mSですから、上位の2～3ビットが点滅していることが確認できるだけです。

上の計算によって、Hレジスタは0．9秒ごとにカウントアップしますから、十分目視することができます。
D、E、Hレジスタの表示結果をもとに計算して、ストップウォッチで計測した結果と比較することもできます。
たとえばD＝01、E＝23、H＝45のときの、スタートしてからの経過時間は次の計算で求められます。

012345は十進数に直すと、74565ですから、
74565×0．92083≒68661．7(秒)になります。
19時間04分21秒です。
ちょっと19時間の観測はきついですから、Dレジスタは00で、EレジスタとHレジスタがカウントアップするのを計測するのがよいでしょう。

[注記]上記の計算の詳細については、当社ホームページの
「つくるCPU[第154回]」(http://www.alles.or.jp/~thisida/mycpu154.html)
を参照してください。

16．タイマールーチンをつくる(NOP命令のテスト)

MYCPU80組立説明書 Ⅲ組立 [9]STA、LDA、STAX、LDAX、XCHG、SPHL、PCHL、INX、DCX、XTHL、SHLD、LHLD、NOP命令回路 の組立作業後に行う動作テストの説明です。

16－1． タイマールーチン(NOP命令のテストプログラム)をメモリに書く

メモリに次のプログラムを書きます(**1． メモリにデータを書き込む** の説明を参考にして操作してください)。

(1)2．5msのタイマールーチン

```
0040  F5          PUSH  PSW  (8)
0041  3EF6        MVI  A, F6   (6)
0043  3D          DCR  A       (8)
0044  C24300      JNZ  $0043  (12)  or  (8)
```

```
0047  00          NOP         （4）
0048  00          NOP         （4）
0049  00          NOP         （4）
004A  00          NOP         （4）
004B  F1          POP  PSW    （8）
004C  C9          RET         （8）
```
（リスト12） 2．5msタイマールーチン（NOP テストプログラム）

（2）0．5secのタイマールーチン

```
0030  F5          PUSH  PSW   （8）
0031  3EC8        MVI  A, C8   （6）
0033  CD4000      CALL  ＄0040 （18＋4962）
0036  3D          DCR  A       （8）
0037  C23300      JNZ  ＄0033  （12） or （8）
003A  F1          POP  PSW     （8）
003B  C9          RET          （8）
```
（リスト13） 0．5secタイマールーチン

CPUクロックは水晶発振ですから、プログラムの実行時間も正確です。
命令を組み合わせることで必要なタイマールーチンを作ることができます。
しかし命令によって、それぞれの実行クロック数が異なりますから、その命令を組み合わせて希望通りのタイマーを作ることは簡単ではありません。
そういうときにNOP命令は、「なにもしないで実行時間だけかかる命令」なので、重宝します。

リスト12はNOPのテストを兼ねていますが、同時に、後々も利用ができるタイマールーチンです。
8ビットのカウンタを使って少ないバイト数で作るタイマールーチンはせいぜいms単位のものになります。
しかし実用上はもう少し長いタイマーがあると便利です。

リスト13はリスト12の2．5msタイマールーチンをCALLすることで作った、0．5secのタイマールーチンです。
リスト12もリスト13も最後がRET命令で終わるサブルーチンですから、単独では実行できません（実行できますが、実行後にRET命令での戻り先がありませんから、暴走してしまいます）。

別に用意するメインルーチンからCALLすることで初めて機能することになります。

命令の後ろの（ ）はその命令の実行クロック数です。
「つくるCPU」のCPUクロックは2MHzですから、1マシンクロックは0．５μsです。
各命令のクロック数に0．５μsを掛けたものが、その命令の実行時間になります。
命令のクロック数は8080命令説明書を参照してください。

16－2． 2．5msタイマールーチンの説明です

0040のPUSH PSWと004BのPOP PSWはスタックにAレジスタとフラグを退避するために使います。
DCR A命令の実行によってAレジスタの値とフラグの状態が変化することをサブルーチンの外に伝えないためです。
このようにPUSH／POPを使うことによってレジスタをサブルーチン内だけのローカルなものにすることができます。

F6は十進数では246になります。
DCR AとJNZが246回繰り返し実行されるほかは、その他の命令が1回ずつ実行されますから、0040からのタイマールーチンの実行クロック数は下のように計算できます。
8＋6＋（8＋12）×246－4＋4＋4＋4＋4＋8＋8＝4962
－4はJNZの最後の一回が8クロックであるための補正です。
1マシンクロックは0．５μsですから、このタイマールーチンの実行時間は4962×0．５＝2481μsになります。
2．5msに少し足りませんが、これは上位のサブルーチン（0．5secタイマールーチン）で正確な2．5msのタイマーにするための補正を考慮しているためです。

16－3.　0. 5secタイマールーチンの説明です

C8は十進数では200になります。
CALLとDCR　AとJNZが200回繰り返し実行されます。
この3つの命令の実行クロック数は18＋8＋12＝38です。
CALLは2. 5msタイマールーチンをCALLしていますから、1回のCALL毎に4962クロックが加わります。
38＋4962＝5000クロックですから、このループを1回実行する毎にちょうど2. 5msかかることになります。
それを200回繰り返しますから、2. 5×200＝500msです。

実際にはこのループの前後にも命令があります。
その部分の実行クロック数を計算します。
8＋6＋8＋8－4＝26クロックです。－4はJNZの最後の1回が4クロック少ないための補正です。
26×0. 5＝13μsです。全体の500msに比べてわずかですから、普通の用途でしたらこのくらいの値は無視できます。

　なおこのタイマールーチンについては、当社ホームページの
「つくるCPU[第157回]」(http://www.alles.or.jp/~thisida/mycpu157.html)
でも説明をしています。

ここで作ったタイマールーチンは次のXCHG命令のテスト他で使います。

17. XCHG命令のテスト

MYCPU80組立説明書　Ⅲ組立　[9]STA、LDA、STAX、LDAX、XCHG、SPHL、PCHL、INX、DCX、XTHL、SHLD、LHLD、NOP命令回路　の組立作業後に行う動作テストの説明です。

17－1.　XCHG命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます(**1.　メモリにデータを書き込む**　の説明を参考にして操作してください)。

```
0000  310000     LXI  SP, $0000
0003  215000     LXI  H, $0050
0006  115100     LXI  D, $0051
0009  010000     LXI  B, $0000
000C  70          MOV  M, B
000D  EB          XCHG
000E  70          MOV  M, B
000F  EB          XCHG
0010  CD3000     CALL  $0030
0013  34          INR  M
0014  46          MOV  B, M
0015  EB          XCHG
0016  CD3000     CALL  $0030
0019  35          DCR  M
001A  4E          MOV  C, M
001B  C30F00     JMP  $000F
```

(リスト14)　XCHG　テストプログラム

ここではXCHG命令の動作テストをするのが目的ですが、リスト14は、それと同時にMOV　MやINR　M、DCR　Mもテストするプログラムになっています。
またアドレス0030の0. 5secタイマールーチンをCALLすることで、NOP命令の動作テストを兼ねています。
[注記]0030からのタイマールーチンがないとリスト14のテストプログラムは実行できません。もしもまだタイマールーチンをメモリに書き込んでいない場合には、16－1. タイマールーチン(NOP命令のテストプログラム)をメモリに書く　の作業をしてください。

このプログラムは、メモリアドレスの0050と0051をレジスタBとレジスタCのワークエリアとして利用し、0. 5秒後にBレジスタ用ワークエリア(0050)をインクリメント(＋1)し、次の0. 5秒後にCレジスタ用ワークエリア(0051)をデクリメント(－1)し、それを繰り返します。

ワークエリアアドレスはHLレジスタを使って指定しますが、HLレジスタに0050が入っているときにもうひとつのアドレス0051はDEレジスタに入れられます。HLレジスタとDEレジスタの値を交換するのにXCHG命令が使われます。

17－2.　テストプログラムを実行する

　プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　プログラムが実行されると、最初Bレジスタと Cレジスタは共に00になり、HLレジスタは0050、DEレジスタは0051になります。
　0．5秒後にBレジスタが＋1され、HLレジスタとDEレジスタの値が交換されます①。
　その次の0．5秒後にCレジスタが－1され、ふたたびHLレジスタとDEレジスタの値が交換されます②。
　①②が繰り返し実行されます。

なおこのXCHGテストプログラムについては、当社ホームページの
　「つくるCPU［第158回］」（http://www.alles.or.jp/~thisida/mycpu158.html）
でも説明をしています。

18.　INX命令のテスト

　MYCPU80組立説明書　Ⅲ組立　［9］STA、LDA、STAX、LDAX、XCHG、SPHL、PCHL、INX、DCX、XTHL、SHLD、LHLD、NOP命令回路　の組立作業後に行う動作テストの説明です。

18－1.　INX命令のテストプログラムをメモリに書く

　メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000     LXI  SP, $0000
0003  210000     LXI  H, $0000
0006  110000     LXI  D, $0000
0009  010000     LXI  B, $0000
000C  CD3000     CALL  $0030
000F  23         INX  H
0010  13         INX  D
0011  03         INX  B
0012  C30C00     JMP  $000C
```

（リスト15）　INX　テストプログラム

　ペアレジスタHL、DE、BCをインクリメントする、INX命令のテストプログラムです。
　0．5secタイマールーチンをCALLしているため、INX　SPはテストから除外しています。
［注記］0030からのタイマールーチンがないとリスト15のテストプログラムは実行できません。もしもまだタイマールーチンをメモリに書き込んでいない場合には、16－1．タイマールーチン（NOP命令のテストプログラム）をメモリに書く　の作業をしてください。

18－2.　テストプログラムを実行する

　プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　プログラムが実行されると、BCレジスタ、DEレジスタ、HLレジスタは0000になり、その後は0．5秒毎にBC、DE、HLレジスタが同時に＋1されていきます。

　なおこのINXテストプログラムについては、当社ホームページの
　「つくるCPU［第159回］」（http://www.alles.or.jp/~thisida/mycpu159.html）
でも説明をしています。

19.　DCX命令のテスト

19－1.　DCX命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000     LXI  SP, $0000
0003  210000     LXI  H, $0000
0006  110000     LXI  D, $0000
0009  010000     LXI  B, $0000
000C  CD3000     CALL  $0030
000F  2B         DCX  H
0010  1B         DCX  D
0011  0B         DCX  B
0012  C30C00     JMP  $000C
```

（リスト16）　DCX　テストプログラム

ペアレジスタHL、DE、BCをデクリメントする、DCX命令のテストプログラムです。
0．5secタイマールーチンをCALLしているため、DCX　SPはテストから除外しています。
[注記]0030からのタイマールーチンがないとリスト16のテストプログラムは実行できません。もしもまだタイマールーチンをメモリに書き込んでいない場合には、16－1．タイマールーチン（NOP命令のテストプログラム）をメモリに書く　の作業をしてください。

19－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、BCレジスタ、DEレジスタ、HLレジスタは0000になり、その後は0．5秒毎にBC、DE、HLレジスタが同時に－1されていきます。

20．STA、LDA、STAX、LDAX命令のテスト

MYCPU80組立説明書　Ⅲ組立　[9]STA、LDA、STAX、LDAX、XCHG、SPHL、PCHL、INX、DCX、XTHL、SHLD、LHLD、NOP命令回路　の組立作業後に行う動作テストの説明です。

20－1.　STA、LDA、STAX、LDAX命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000     LXI  SP, $0000
0003  210000     LXI  H, $0000
0006  115100     LXI  D, $0051
0009  015000     LXI  B, $0050
000C  CD3000     CALL  $0030
000F  23         INX  H
0010  7D         MOV  A, L
0011  325200     STA  $0052
0014  3A5200     LDA  $0052
0017  12         STAX  D
0018  1A         LDAX  D
0019  02         STAX  B
001A  0A         LDAX  B
001B  6F         MOV  L, A
001C  C30C00     JMP  $000C
```

（リスト17）　STA、LDA、STAX、LDAX　テストプログラム

STA、LDA、STAX、LDAXの各命令のテストプログラムです。
いずれもAレジスタの値をメモリにSTORE、またはメモリからLOADする命令です。
STA、LDAはSTORE、LOADするメモリアドレスを直接指定します。
STAX、LDAXはメモリアドレスをBCレジスタ、DEレジスタで指定する「間接アドレッシング」命令です。
HLレジスタを＋1したあと、Lレジスタの値をAレジスタに入れます。
Aレジスタの値をSTA命令でアドレス0052に書き込み、すぐにLDA命令で読み出します。
次にSTAX　DとLDAX　Dで同じようにDEレジスタで示すメモリアドレス（0051）にAレジスタの値を書き込み、読み出しし、その次にSTAX　BとLDAX　BでBCレジスタで示すメモリアドレス（0050）にAレジスタの値を書き込み、読み出しし、その値をLレジスタに戻します。
この動作を0．5秒ごとに繰り返します。
［注記］0030からのタイマールーチンがないとリスト17のテストプログラムは実行できません。もしもまだタイマールーチンをメモリに書き込んでいない場合には、16－1．タイマールーチン（NOP命令のテストプログラム）をメモリに書く の作業をしてください。

20－2．　テストプログラムを実行する

プログラムの実行の仕方については　**4．　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、BCレジスタは0050、DEレジスタは0051、HLレジスタは0000になり、その後は0．5秒毎にHLレジスタの値が＋1されていきます。Aレジスタはタイマールーチンの中で高速でデクリメントを続けますから、全点灯しているようにしか見えません。

なおこのSTA、LDA、STAX、LDAXテストプログラムについては、当社ホームページの
「つくるCPU［第160回］」（http://www.alles.or.jp/~thisida/mycpu160.html）でも説明をしています。

21．SHLD、LHLD、XTHL命令のテスト

MYCPU80組立説明書　Ⅲ組立　［9］STA、LDA、STAX、LDAX、XCHG、SPHL、PCHL、INX、DCX、XTHL、SHLD、LHLD、NOP命令回路　の組立作業後に行う動作テストの説明です。

21－1．　SHLD、LHLD、XTHL命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000     LXI  SP, 0000
0003  010000     LXI  B, 0000
0006  110000     LXI  D, 0000
0009  3E00       MVI  A, 00
000B  C5         PUSH  B     (8)
000C  E3         XTHL        (16)
000D  23         INX  H      (8)or(12)
000E  225000     SHLD  0050  (20)
0011  210000     LXI  H, 0000  (8)
0014  2A5000     LHLD  0050  (20)
0017  E3         XTHL        (16)
0018  C1         POP  B      (8)
0019  1C         INR  E      (8)
001A  C20B00     JNZ  000B   (12)or(8)
001D  14         INR  D      (8)
001E  C20B00     JNZ  000B   (12)or(8)
0021  3C         INR  A      (8)
0022  C30B00     JMP  000B   (12)
```

（リスト18）SHLD、LHLD、XTHL　テストプログラム

SHLD、LHLD、XTHLの各命令のテストプログラムです。
いずれもHLレジスタとメモリとの間でデータを交換する命令です。

SHLD、LHLDはHLの値をメモリにSTORE、メモリからLOADします。

XTHLはスタックのトップにある2バイトのデータとHLの値とを交換する命令です。

BCレジスタとDEレジスタには初期値0000を入れます。
Aレジスタにも00を入れます。

PUSH　B、POP　Bは、XTHLの動作を確認するために使っています。
PUSH　B命令の実行によって、スタックのトップにはBCレジスタの値が置かれます。
XTHLを実行すると、そのときにスタックトップにある2バイトのデータと、HLレジスタの値とが交換されます。
XTHL命令によって、スタックトップのデータ（実はBCレジスタの値と同じ）がHLレジスタに入り、代わりにHLレジスタの値がスタックに置かれます。
BCレジスタは変化しませんが、動作としては、BCとHLの値を交換したに近い動作になります。
ですから、HLの値は、実はBCの値と同じになっています。

そのHLの値を次のINX　Hで＋1します。
そしてSHLDとLHLDのテストのために、まずSHLD命令で、メモリの0050にHLの値をSTOREします。
その後に、LHLD命令で、メモリの0050から、ふただび値をHLレジスタに読み込みます。
そしてもう一度、XTHLを実行して、スタックトップのデータ（保存してあった、もともとのHLのデータ）と、現在のHLレジスタの値とを交換します。

ここまでの動作で、スタックトップの値（実はBCレジスタの値）が＋1されたことになります。
そして最後に、POP　Bで、スタックトップの値をBCレジスタに戻します。
結局、HLレジスタの値は変わることなく、BCレジスタの値だけが＋1されたことになります。

実行時間を時計で計って、計算結果と比較できるように、以上の動作を1回実行するごとに、Eレジスタを＋1します。
Eレジスタを＋1した結果が00でなければ、000Bに戻って繰り返します。
結果が00のときは、Dレジスタを＋1します。

Dレジスタを＋1した結果が00でなければ000Bに戻って繰り返します。
Dレジスタを＋1した結果が00のときは、Aレジスタを＋1してから、000Bに戻って繰り返します。

21－2.　テストプログラムを実行する

　プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　プログラムが実行されると、DEレジスタは0000、Aレジスタは00になり、A、D、Eをつないだ24ビットカウンタとしてカウントアップしていきます。同時にBCレジスタもカウントアップしていきます。

　カウントアップされていく時間を計算してみます。
　リスト18の各命令の後ろに（）で示されているのが、その命令の実行クロック数です。
　必ず繰り返し実行されるのは、000B～001Cです。
　そこで、この範囲の命令が1回実行されるのに必要な時間をまず求めてみます。

　000DのINX　Hと、001AのJNZ　000Bには実行クロック数が2つあります。
INX　Hは、実行した結果、下位8ビットが00になるときだけ12クロックで、それ以外のときは8クロックです。
JNZ　000Bは、Zフラグが立っているときだけ（つまりその前に実行されるINR　Eの実行の結果、Eが00になったときだけ）8クロックで、それ以外のときは、12クロックです。

　プログラムは、B、C、D、E、Aに00を初期セットするところからスタートします。
　H、Lには何がはいっているかわかりませんが、000B～001Cの中では、B、Cの値がH、Lに入ることになりますから、H、Lも最初は00からスタートすることになります。
　INX　HとINR　Eは同じように、00からスタートして＋1ずつインクリメントされていきますから、Lレジスタが00になるタイミングとEレジスタが00になるタイミングは同じになります。
　ということは、000DのINX　Hと、001AのJNZ　000Bの2つの命令の実行クロック数の合計は、常に20クロックということになります。

その20クロックに残りの命令の実行クロック数を加算すると、合計は124クロックになります。
1マシンクロックは0．5μsですから124×0．5＝62μsです。
つまり、000B～001Cが1回実行されて、HLレジスタ（実はBCレジスタ）とEレジスタが＋1されるのに必要な実行時間は62μsということです。
そして、Eレジスタが00からはじまって、また00になるまで、256カウントするたびに、Dレジスタが＋1されます。

そこで今度は、Dレジスタが＋1されるのにかかる時間を計算してみます。
Eレジスタが256カウントするのにかかる時間は、
62×256＝15872μsです。
INR Dとその次のJNZ 000Bの実行クロック数は8＋12＝20ですから実行時間は10μsです。
これを加算すると、
15872＋10＝15882μsになります。
これがDレジスタが＋1されるのにかかる時間です。

同様にして、Aレジスタが＋1されるのにかかる時間を計算します。
Dレジスタが256カウントするたびに、Aレジスタが＋1されるのですから、
15882×256－2＋10＝4065800μsになります。
最後の10はINR Aとその次のJMP 000Bの実行時間です。
その前の－2は、Dレジスタが256カウントする間に、1回だけ結果が00になって、JNZ命令の条件が不成立になるので、残りの255回の実行時間6μsより2μs短くなることによる差を引いているものです。

以上の計算から、Aレジスタは4．0658秒ごとに＋1されていくことがわかります。

なおこのSHLD、LHLD、XTHL テストプログラムについては、当社ホームページの
「つくるCPU［第161回］」（http://www.alles.or.jp/~thisida/mycpu161.html）
でも説明をしています。

22．SPHL、PCHL命令のテスト

MYCPU80組立説明書 Ⅲ組立 ［9］STA、LDA、STAX、LDAX、XCHG、SPHL、PCHL、INX、DCX、XTHL、SHLD、LHLD、NOP命令回路 の組立作業後に行う動作テストの説明です。

22－1． SPHL、PCHL命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1． メモリにデータを書き込む** の説明を参考にして操作してください）。

```
0000 3E00      MVI A, 00
0002 47        MOV B, A
0003 4F        MOV C, A
0004 57        MOV D, A
0005 5F        MOV E, A
0006 210001    LXI H, $0100
0009 F9        SPHL        (8)
000A E9        PCHL        (8)

00FE 00        DB 00
00FF 00        DB 00
0100 3B        DCX SP      (8)or(12)
0101 3B        DCX SP      (8)or(12)
0102 F1        POP PSW     (8)
0103 3C        INR A       (8)
0104 F5        PUSH PSW    (8)
0105 C20901    JNZ $0109    (12)or(8)
0108 03        INX B       (8)or(12)
0109 24        INR H       (8)
010A E5        PUSH H      (8)
010B 210900    LXI H, $0009 (8)
```

```
010E  E3          XTHL          (16)
010F  C9          RET           (8)

01FE  00          DB  00
01FF  00          DB  00
0200  3B          DCX  SP       (8)or(12)
0201  3B          DCX  SP       (8)or(12)
0202  F1          POP  PSW      (8)
0203  3C          INR  A        (8)
0204  F5          PUSH  PSW     (8)
0205  C20902      JNZ  $0209    (12)or(8)
0208  13          INX  D        (8)or(12)
0209  25          DCR  H        (8)
020A  C30A01      JMP  $010A    (12)
```

(リスト19)　SPHL、PCHL　テストプログラム

SPHL命令とPCHL命令のテストプログラムです。
SPHLはHLレジスタの値をSP(スタックポインタ)に転送する命令です。
PCHLはPC(プログラムカウンタ)にHLの値を転送します。PC(プログラムカウンタ)がHLの値になりますから、HLの値で示すアドレスにジャンプするという動作になります。

リスト19は2つの24ビットカウンタB、C、AとD、E、Aを交互にカウントアップさせる動作をします。
Aレジスタが共用されているため、スタックに値を退避します。
SPHLとPCHLを使って2組のプログラムと2組のスタックを使い分けています。

HLレジスタには初期値として0100を入れています。
この0100は「BCA」カウンタルーチンのエントリアドレスであるとともに、スタックポインタのアドレスでもあります。

次の命令、SPHLで、SP(スタックポインタ)に、この0100がセットされます。
そして、さらにその次の、PCHL命令によって、PC(プログラムカウンタ)にも同じ0100がセットされます。
PCの値が0100になりますから、つまり、これはJMP　0100が実行されるのと同じことです。
したがって、処理の流れは、0100に移ります。

0100と0101にはDCX　SP命令があります。
DCX　SPを2回実行しますから、SPの値は0100－2で00FEになります。
次にPOP　PSWを実行します。
SP(スタックポインタ)に00FEをセットしておいて、POP　PSWを実行することで、00FE～00FFのメモリ内容がAレジスタとフラグレジスタに入れられます。
POP　PSWのあと、INR　Aを実行して、Aレジスタを＋1(インクリメント)します。
そして、PUSH　PSW命令で、＋1したあとのAレジスタの値をまたスタックに保存します。

Aレジスタが256カウントされて、00になるたびに、その上位カウンタのBCレジスタを＋1します。
次にINR　Hを実行します。Hレジスタが＋1されます。
HLレジスタには「BCA」カウンタルーチンを実行するために、そのエントリアドレスの0100が入っていました。
そのHレジスタが＋1されるとHLレジスタの値は、0200になります。

0200はもうひとつのカウンタ「DEA」をインクリメントするためのルーチンのエントリアドレスです。
0200～のメモリアドレスには、0100～と同じ動作をするプログラム(対象になるレジスタだけが異なっている)が書かれています。

0100～のプログラムを実行したあとで、次に実行するプログラムのエントリアドレスである0200をHLレジスタに入れた後、PUSH　Hを実行し、最後にRETを実行します。
こうすることで、0200にジャンプしてもうひとつのカウンタルーチンが実行されることになります。

このようにして、0100～のプログラムと0200～のプログラムが交互に実行され、B、C、AレジスタとD、E、Aカウンタが交互にインクリメントされることを繰り返します。

22－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、DEレジスタとBCレジスタは0000からカウントアップを開始します。
Aレジスタもカウントアップを続けますが速過ぎてLEDは全点灯しているようにしか見えません。
HLレジスタには0100、0200、0009が高速で繰り返し書き込まれるため、LED表示はそれらの値が合成されて見えます。

カウントアップされていく時間を計算してみます。
リスト19の各命令の後ろに()で示されているのが、その命令の実行クロック数です。

まずはプログラムの流れから見ていきます。
繰り返し実行される部分は0009から後ろです。
0100～と0200～はカウントアップされるレジスタがBCであるかDEであるかが違っているだけであとは変わりませんから、0100～のみの流れを見ていくことにします。

SPHL、PCHLが実行されたあとは、0100のDCX　SPから0105のJNZまでは毎回実行されます。
JNZはその前のINR　Aの結果によって分岐しますが、Zフラグが0NになるのはAレジスタか256回インクリメントされるうちでA＝00になる1回だけで、残りの255回はZフラグはOFFですから、ここは毎回0109へのジャンプが実行される、と考えます(INX　Bがパスされる)。
その後は0109のINR　Hから010FのRETまでが実行されてから0009に戻ります。

以上の流れの実行クロック数を計算します。

8＋8＋12[＊注]＋8＋8＋8＋8＋12＋8＋8＋8＋16＋8＝120

[＊注]0101のDCX　SPは8クロックなのに0100のDCX　SPは12クロックになっています。
DCX命令は下位レジスタから上位レジスタへのボローが発生するときだけ12クロックで、それ以外は8クロックです。
0100のときのSPには0100が入っていてこれを－1しますから、必ずボローが発生します(0100→00FF)。それが0100のDCX　SPのクロック数が12になっている理由です。

120クロックでAレジスタが＋1されます。
Aレジスタが256回インクリメントされる度に、BCレジスタが＋1されます。

BCレジスタが＋1される時間を求めてみます。
実行クロック数は
120×256－4＋8＝30724クロック
になります(－4は256回に1回JNZが8クロックになるための補正です。＋8はINX　Bの実行クロックです)。
じつは、BCレジスタは30724クロックごとにインクリメントされるのではなくて、0100～のBCレジスタのインクリメントルーチンと0200～のDEレジスタのインクリメントルーチンは交互に実行されます。
したがってBCレジスタのインクリメントは(DEレジスタのインクリメントも同じ)、0100～と0200～の実行クロック数の合計ごとに行われることになります。
0200～のルーチンは最後のJMP　$010Aのクロック数だけ余分にかかります。
ですから0200～の実行クロック数は
(120＋12)×256－4＋8＝33796になります。
したがってBCレジスタまたはDEレジスタが＋1されるときの実行クロック数は、30724＋33796＝64520クロックになります。

1クロックは0．5μsですから、実行時間は
64520×0．5＝32260μs　になります。

約32msごとにCレジスタ(またはEレジスタ)がカウントアップされるので、LED表示の様子は、上位ビットは点滅して見えても下位数ビットは全点灯しているようにしか見えません。
そこで、次にBレジスタ(Dレジスタ)が＋1される時間を計算してみます。
Cレジスタ(Eレジスタ)が256カウントされるごとにBレジスタ(Dレジスタ)が＋1されるのですから、その実行時間は、

32．260×256＋0．002×2＝8258．564msになります。
＋0．002×2は、Bレジスタ（Dレジスタ）が＋1されるときは、Cレジスタ（Eレジスタ）からの繰り上がりが発生しているわけですから、そのときのINX命令で余計にかかる2μsです。交互に1回ずつの合計ですから、その2倍です（INXもDCXと同じように、下位レジスタから上位レジスタへの繰り上がりがあるときは4クロック長くなります）。

以上の計算から、Bレジスタ（Dレジスタ）は約8秒ごとに＋1されます。

なおこのSPHL、PCHL　テストプログラムについては、当社ホームページの
「つくるCPU［第163回］」（http://www.alles.or.jp/~thisida/mycpu163.html）～「つくるCPU［第165回］」
（http://www.alles.or.jp/~thisida/mycpu165.html）
でも説明をしています。

23．IN、OUT命令のテスト

MYCPU80組立説明書　Ⅲ組立　［10］IN、OUT、RLC～RAR、STC、CMC、CMA命令回路　の組立作業後に行う動作テストの説明です。

23－1．　テスト用の回路を準備する

IN命令は外部I／O回路から8ビットのデータを入力します。
OUT命令は外部I／O回路に8ビットのデータを出力します。
動作はメモリに対するREAD／WRITEとよく似ています。
メモリに対するMOV命令などでは外部アドレスバスA15～A0にメモリアドレスを出力し、データバスD7～D0を使ってデータの読み書きをします。
これに対してIN、OUT命令はアドレスバスの下位8ビットA7～A0にI／Oアドレスを出力し、データバスD7～D0を使ってデータの入出力を行います。
アドレスバスA7～A0はメモリ回路と共通していますから、そのままではメモリ回路も同時にアクセスされてしまいます。
そうならないように、IN、OUT命令ではメモリ制御信号のMEMRD、MEMWRを使う代わりに、IORD、IOWR信号をアクティブにします。

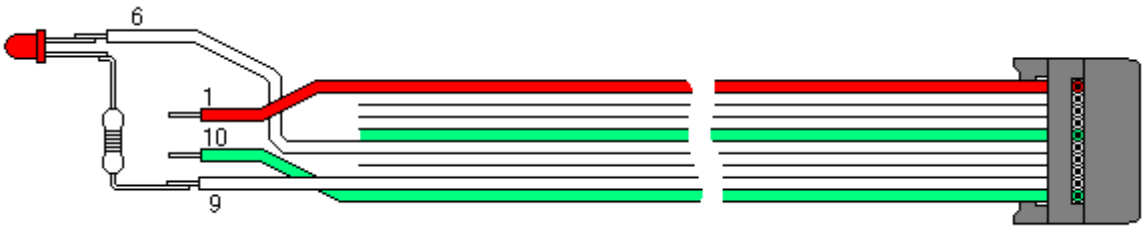
IN、OUT命令回路のテストをするためには、A7～A0とIORD、IOWRによってセレクトされるI／O入出力回路が必要です。
MYCPU80は基板上にそのための回路を実装しているので、外部にスイッチやLEDを接続するだけで簡単にIN、OUTの動作を確認することができます。

回路についてはMYCPU回路図№31、№32を参照してください。
I／Oアドレス98～9Bでアクセスされる入出力信号端子がCN3の10pinコネクタに配置してあります。

CN3端子接続図

IN7	1	2	IN6
IN5	3	4	IN4
IN3	5	6	OUT7
OUT6	7	8	OUT5
＋5V	9	10	GND

CN3には10pフラットケーブルを接続します。
その前に10pフラットケーブルのコネクタがついていない側にテスト用のLEDをとりつけます。

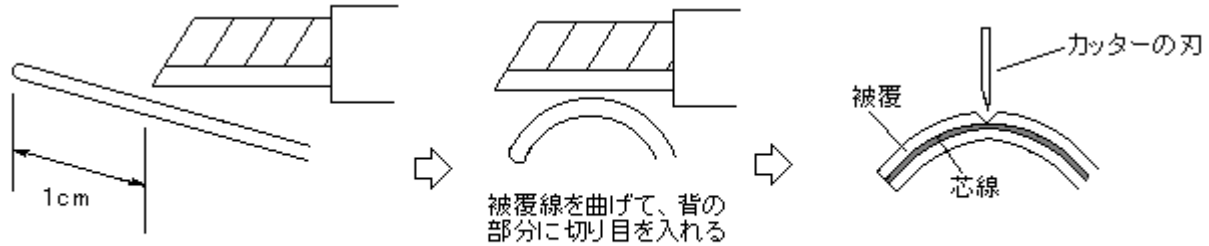


フラットケーブルの赤い色がついている側が1番pinです。5番、10番は緑色です。

フラットケーブルは隣同士のケーブル被覆が10cmごとにくっついていますが指先でつまんで引っ張るようにすると1本ずつばらばらにすることができます。
図のように1番、6番、9番、10番の被覆を1cmほどはがして(＊1)、できればハンダあげ(＊2)をしておきます。
赤色LEDの長い側の端子に2．2KΩの抵抗をハンダづけします。
抵抗のもう一方のリード線にフラットケーブルの9番をハンダづけします。
赤色LEDの短い側の端子にフラットケーブルの6番をハンダづけします。
フラットケーブルの1番と10番は図のように何もつながない状態にしておきます。

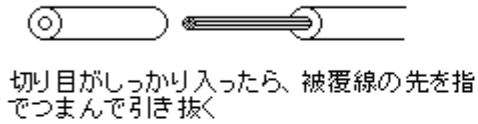
[注意]フラットケーブルの9番は＋5V、10番はGNDです。この2本が他の端子や基板配線などに接触しないように注意してください。とくに9番と10番が接触すると電源がショートしますので十分に注意してください。

(＊1)ビニール被覆線の被覆のむきかた



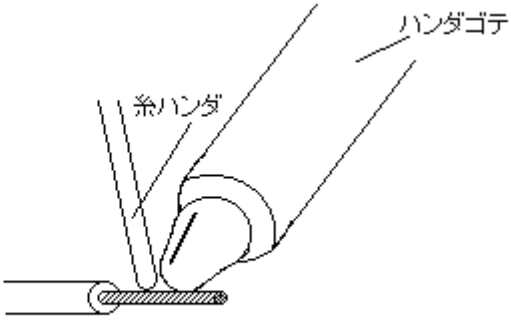
ビニール被覆線の先端から1cmていどのところに、カッターナイフで切り目を入れる

(＊2)ハンダあげ



切り目がしっかり入ったら、被覆線の先を指でつまんで引き抜く

芯線はよくよじっておく



芯線にハンダを吸わせるようにしてハンダをつけておく

23－2．　フラットケーブルコネクタをCN3に接続する

以上のように作業した10pフラットケーブルコネクタをCN3に接続します。コネクタの△マークと基板面のシルク印刷の△マークを合わせる向きで接続します。このとき10本のコネクタ端子がずれたりしないで全部フラットケーブルコネクタにおさまっていることを確認してください。

23－3．　IN、OUT命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます(1．　**メモリにデータを書き込む**　の説明を参考にして操作してください)。

```
0000  DB98       IN  98
0002  D398       OUT  98
0004  C30000    JMP  $0000
```

(リスト20)IN、OUT　テストプログラム

たったこれだけの非常に短いプログラムです。
I／Oアドレス98から入力したデータ(Aレジスタに読み込まれます)を、I／Oアドレス98に出力します。
同じアドレスですがINとOUTで回路は別になっていますから、入力ラインに出力されることはありません。

23－4.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、IN　98命令でI／Oアドレスの98からデータがAレジスタに入力され、その後OUT　98命令でAレジスタに入力されたデータがI／Oアドレス98に出力されます。そしてそれを繰り返します。

CN3に接続された10pフラットケーブルの1番はI／Oアドレス98のビット7入力です。
また赤LEDを接続した6番はI／Oアドレス98のビット7出力です。
LEDはビット7出力がHのとき消灯し、Lのとき点灯します。

プログラムを実行させたまま、10pフラットケーブルの1番と10番をショートさせてみてください。
10番はGNDですが、1番は入力信号端子ですからGNDとショートさせても構いません。
するとI／O入力のビット7がGNDにショートしてLレベル（0）になりますから、そしてその他のビット入力は抵抗でプルアップされていてHレベル（1）が入力されますから、Aレジスタには7Fが入力されます。
AレジスタのLEDは左端のビット7だけが消灯します。
同時にそのデータがOUTアドレスの98に出力されるため、フラットケーブルに接続した赤LEDが点灯します。
そしてフラットケーブルの1番と10番を離すと1番の入力も抵抗でプルアップされているのでH（1）入力となり、Aレジスタには FFが入力されます。フラットケーブルにつないだ赤LEDは消灯します。

なおIN命令、OUT命令については、当社ホームページの
「つくるCPU［第166回］」（http://www.alles.or.jp/~thisida/mycpu166.html）
でも説明をしています。

24. OUT命令のテスト

MYCPU80組立説明書　Ⅲ組立　［10］IN、OUT、RLC～RAR、STC、CMC、CMA命令回路　の組立作業後に行う動作テストの説明です。

24－1.　OUT命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000　3C　　INR　A（8）
0001　D398　　OUT　98（8）
0003　C30000　JMP　$0000（12）
```
（リスト21）OUT　テストプログラム

Aレジスタをインクリメント（＋1）して、その結果をOUTアドレス98から出力します。
これを繰り返します。

24－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、Aレジスタがインクリメント（＋1）され続けて、その値がOUTアドレス98から継続して出力されます。
OUTアドレス98の各ビットからの出力は、Aレジスタの各ビットの値と同じですから、それぞれの値のパルス出力になります。

23－1.　テスト用の回路を準備する　で作成し、CN3に接続した10pフラットケーブルの6番（赤LEDが接続されています）から出力されるパルスを計算してみます。
リスト21の各命令の後ろに（）で示されているのが、その命令の実行クロック数です。
Aレジスタが＋1され、OUTアドレスから出力される繰り返しの1回あたりのクロック数は、下の計算で求められます。

8＋8＋12＝28クロック

10pフラットケーブルの6番はOUTアドレスのビット7で、それはそのままAレジスタのビット7ですから、128回の繰り返しごとに1と0が交互に出力されます（00～7Fのとき出力は0、80～FFのとき出力は1）。

28×128＝3584クロック
1クロックは0．5μsですから、3584×0．5＝1792μsです。
10pフラットケーブルの6番からは1．792msごとにHとLが交互に出力されることになります。
この出力をオシロスコープにつないで見ると、HとLがともに1．792msの方形波（周期は3．584ms）が観測できます。
周波数カウンタがあれば、1／3．584≒0．279ですから279Hzの測定値が得られます。

24－3．　オシロスコープや周波数カウンタを使わないで観測する

次のプログラムをメモリに書いて、実行してください。

```
0000  04          INR  B(8)
0001  C20000     JNZ  $0000(12  or  8)
0004  3C          INR  A(8)
0005  D398        OUT  98(8)
0007  C30000     JMP  $0000(12)
```
(リスト22)OUT　テストプログラム(2)

実行クロック数を計算します。
Bレジスタが256回インクリメント（＋1）されるごとにAレジスタが＋1されます。
Aレジスタが＋1されるときのクロック数をまず求めます。

（8＋12）×256－4＋8＋8＋12＝5144クロック

－4は256回に1回、Bレジスタが00になったときだけJNZ命令が8クロックになることによる補正です。
これが128回繰り返されるごとに10pフラットケーブルの6番からの出力が反転します。

5144×128＝658432クロック
658432×0．5＝329216μs

0．33秒ごとに10pフラットケーブルの6番に取り付けた赤LEDが点滅するのが確認できます。
AレジスタのLEDもインクリメントするのが観測できます。下位ビットは点滅が速いので（最下位ビットは2．5ms毎に点滅）なので全点灯しているようにしか見えませんが、上位ビットは見分けが可能です。最上位のビット7は10pフラットケーブルに取り付けた赤LEDも同じ周期で点滅することがわかりますが、点灯、消灯の状態は逆になっています。AレジスタのLEDはビットが1のときに点灯するのに対し、10pフラットケーブルに取り付けた赤LEDは0のときに点灯するようになっているからです。

25．　RLC～RAR命令のテスト

MYCPU80組立説明書　Ⅲ組立　[10]IN、OUT、RLC～RAR、STC、CMC、CMA命令回路　の組立作業後に行う動作テストの説明です。

25－1．　RLC命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000    LXI  SP, $0000
0003  3E80       MVI  A, 80
0005  CD3000    CALL  $0030
0008  07         RLC
0009  4F         MOV  C, A
000A  C30500    JMP  $0005
```
(リスト23)RLC　テストプログラム

[注記]0030からのタイマールーチンがないとリスト23のテストプログラムは実行できません。もしまだタイマールーチンをメモリに書き込んでいない場合には、16－1．タイマールーチン（NOP命令のテストプログラム）をメモリに書くの作業をしてください。
組立説明書か、この操作説明書にしたがって、前から順にプログラムのテストをしてきた場合には、RAMに書かれたプログラムやデータはボタン電池でバックアップされていますから、0030からのタイマールーチンも残っているはずですが念の為に確認してみてください。

RLCはAレジスタを左に（ビット0からビット7の方向に）1ビットシフトする命令です。
ビット7の値はビット0に入れられますが、同時にC（キャリー）フラグにも入れられます。
リスト23はビット7だけが1のデータ（80）を初期値としてAレジスタに入れた後、0．5秒ごとに左シフトすることを繰り返すものです。
0030からのタイマールーチンの中でAレジスタが使われているため、Aレジスタがシフトされる様子をLEDで確認することはできません。MOV　C，Aはそのためにつけ加えられたものです。
Aレジスタの様子はLEDで確認することができませんが、代わりにCレジスタのLEDでその様子を見ることができます。

25－2．　テストプログラムを実行する

プログラムの実行の仕方については　**4．　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、0．5秒ごとにCレジスタの値が80→01→02→04→08→10→20→40→80の順に左シフトします。80から01になるタイミングでC（キャリー）フラグが点灯しますが、その他のタイミングでは消灯します。

リスト23の0008　07　RLCを、0F（RRC）、17（RAL）、1F（RAR）に書き換えることで、それぞれの命令の動作を確認することができます（各命令の動作については8080命令説明書を参照してください）。

なおRLC～RAR命令については、当社ホームページの
「つくるCPU［第168回］」（http://www.alles.or.jp/~thisida/mycpu168.html）～「つくるCPU［第171回］」
（http://www.alles.or.jp/~thisida/mycpu171.html）
でも説明をしています。

26．　STC、CMC、CMA命令のテスト

MYCPU80組立説明書　Ⅲ組立　［10］IN、OUT、RLC～RAR、STC、CMC、CMA命令回路　の組立作業後に行う動作テストの説明です。［訂正］CMA命令は［11］の作業後でなければ機能しないことを見落としていました。このテストは［11］の組立作業が終わってから行ってください。

26－1．　STC、CMC、CMA命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  310000    LXI  SP，＄0000
0003  3E00      MVI  A，00
0005  4F        MOV  C，A
0006  07        RLC
0007  CD3000    CALL  ＄0030
000A  CD3000    CALL  ＄0030
000D  37        STC
000E  CD3000    CALL  ＄0030
0011  2F        CMA
0012  4F        MOV  C，A
0013  3F        CMC
0014  C30E00    JMP  ＄000E
```

（リスト24）RLC　テストプログラム

[注記]0030からのタイマールーチンがないとリスト24のテストプログラムは実行できません。もしまだタイマールーチンをメモリに書き込んでいない場合には、16－1．タイマールーチン（NOP命令のテストプログラム）をメモリに書くの作業をしてください。
組立説明書か、この操作説明書にしたがって、前から順にプログラムのテストをしてきた場合には、RAMに書かれたプログラムやデータはボタン電池でバックアップされていますから、0030からのタイマールーチンも残っているはずですが念の為に確認してみてください。

STCはC（キャリー）フラグをセットする命令です。
CMCはC（キャリー）フラグを反転させる命令です。
CMAはAレジスタの値を反転させる（1のビットを0に、0のビットを1にする命令です。
リスト24はC（キャリー）フラグとAレジスタの値を、0．5秒ごとに反転させることを繰り返すものです。
0030からのタイマールーチンの中でAレジスタが使われているため、Aレジスタが反転される様子をLEDで確認することはできません。MOV　C，Aはそのためにつけ加えられたものです。
Aレジスタの様子はLEDで確認することができませんが、代わりにCレジスタのLEDでその様子を見ることができます。

26－2．　テストプログラムを実行する

プログラムの実行の仕方については　**4．　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムが実行されると、最初1秒間はCレジスタとC（キャリー）フラグが0クリアされ、LEDが消灯します。その後は0．5秒ごとにCレジスタの値が00とFFを交互に繰り返し、また同じタイミングでC（キャリー）フラグが点滅します。

なおSTC、CMC、CMA命令については、当社ホームページの
「つくるCPU[第167回]」（http://www.alles.or.jp/~thisida/mycpu168.html）、
「つくるCPU[第170回]」（http://www.alles.or.jp/~thisida/mycpu170.html）、
「つくるCPU[第171回]」（http://www.alles.or.jp/~thisida/mycpu171.html）
でも説明をしています。

27．　ANA、XRA、ORA、ANI命令のテスト

MYCPU80組立説明書　Ⅲ組立　[11]ANA、XRA、ORA、ANI、XRI、ORI命令回路　の組立作業後に行う動作テストの説明です。

27－1．　ANA、XRA、ORA、ANI命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1．　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  3E91     MVI  A, 91
0002  E6F0     ANI  F0
0004  4F       MOV  C, A
0005  3E91     MVI  A, 91
0007  06F0     MVI  B, F0
0009  B0       ORA  B
000A  57       MOV  D, A
000B  3E91     MVI  A, 91
000D  A8       XRA  B
000E  5F       MOV  E, A
000F  210002   LXI  H, $0200
0012  36F0     MVI  M, F0
0014  A6       ANA  M
0015  6F       MOV  L, A
0016  AF       XRA  A
0017  76       HLT
```

（リスト25）ANA、XRA、ORA、ANI　テストプログラム

リスト25はANA、XRA、ORA、ANI命令のテストプログラムです。
XRIとORIについてはテストしませんが、ANA～ORIは基本的な部分はすべて同じ回路を共有していますから、ここでのテストは省略します。

27－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムは一瞬で実行され、最後のHLT命令が実行され続けられます。
最初にAレジスタの値91と定数F0のANDが計算され、結果の値90がCレジスタに入れられます。
次にAレジスタの値91とBレジスタの値F0のORが計算され、結果の値F1がDレジスタに入れられます。
次にAレジスタの値91とBレジスタの値F0のXORが計算され、結果の値61がEレジスタに入れられます。
次にAレジスタの値61とメモリアドレス0200の値F0のANDが計算され、結果の値60がLレジスタに入れられます。
最後にXOR　Aが実行され、その結果Aレジスタがクリアされて00になります。
以上が実行されたあとHLT命令が実行され続けられます。

なおANA、XRA、ORA命令については、当社ホームページの
「つくるCPU［第179回］」（http://www.alles.or.jp/~thisida/mycpu179.html）、
「つくるCPU［第184回］」（http://www.alles.or.jp/~thisida/mycpu184.html）、
でも説明をしています。

28. ADD、ADC、SUB、SBB、CMP命令のテスト

MYCPU80組立説明書　Ⅲ組立　［12］ADD、SUB、CMP、DAD、DAA、RST、INT命令回路　の組立作業後に行う動作テストの説明です。

28－1.　ADD、ADC、SUB、SBB、CMP命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  21AB56    LXI  H, $56AB
0003  118934    LXI  D, $3489
0006  7D        MOV  A, L
0007  83        ADD  E
0008  4F        MOV  C, A
0009  7C        MOV  A, H
000A  8A        ADC  D
000B  47        MOV  B, A
000C  11CD12    LXI  D, $12CD
000F  3E89      MVI  A, 89
0011  93        SUB  E
0012  6F        MOV  L, A
0013  3E34      MVI  A, 34
0015  9A        SBB  D
0016  67        MOV  H, A
0017  BC        CMP  H
0018  C21D00    JNZ  $001D
001B  BD        CMP  L
001C  76        HLT
001D  76        HLT
```

（リスト26）ADD、ADC、SUB、SBB、CMP　テストプログラム

リスト26はADD、ADC、SUB、SBB、CMP命令のテストプログラムです。
ADI、ACI、SUI、SBI、CPIについてはテストしませんが、ここでテストするADD～CMPと基本的な部分はすべて同じ回路を共有していますから、ここでのテストは省略します。

28－2.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムは一瞬で実行され、最後のHLT命令が繰り返し実行されます。
最初にAレジスタの値ABとEレジスタの値89が加算（ADD）され、結果の値34がCレジスタに入れられます。
このときキャリーが発生します。
そのキャリーとAレジスタの値56とDレジスタの値34とが加算（ADC）され、結果の値8BがBレジスタに入れられます。
次にAレジスタの値89からEレジスタの値CDが減算（SUB）され、結果の値BCがLレジスタに入れられます。
このときボローが発生します。
次にAレジスタの値34からDレジスタの値12とボローとが減算（SBB）され、結果の値21がHレジスタに入れられます。
そのあとAレジスタとHレジスタが比較（CMP）されます。
Aレジスタの値21がHレジスタに入れられた直後ですから、A＝HでZフラグが1になります。
したがってその次のJNZはパスされて、その次のCMP命令が実行されます。
Aレジスタの値21とLレジスタの値BCが比較されます。CMP命令の比較はSUBと同じ計算をしますから、21－BCの計算になります。その結果キャリーフラグが立ちます。
最後にHLT命令が繰り返し実行されます。
プログラムが正しく実行されれば、B＝8B、C＝34、A＝21、H＝21、L＝BCになりC（キャリー）フラグがセットされたあと、アドレス001CのHLT命令で停止（繰り返し実行）されます。

なおADD、ADC、SUB、SBB、CMP命令については、当社ホームページの
「つくるCPU［第190回］」（http://www.alles.or.jp/~thisida/mycpu190.html）～「つくるCPU［第205回］」
（http://www.alles.or.jp/~thisida/mycpu205.html）、
でも説明をしています。

29. DAD命令のテスト

MYCPU80組立説明書　Ⅲ組立　［12］ADD、SUB、CMP、DAD、DAA、RST、INT命令回路　の組立作業後に行う動作テストの説明です。

29－1.　DAD命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  012301    LXI  B, $0123
0003  116745    LXI  D, $4567
0006  21AB89    LXI  H, $89AB
0009  31EFCD    LXI  SP, $CDEF
000C  09        DAD  B
000D  44        MOV  B, H
000E  4D        MOV  C, L
000F  19        DAD  D
0010  54        MOV  D, H
0011  5D        MOV  E, L
0012  39        DAD  SP
0013  F9        SPHL
0014  29        DAD  H
0015  76        HLT
```

（リスト27）DAD　テストプログラム

DAD命令はHLレジスタとBC、DE、HL、SPとを加算して結果をHLレジスタに入れる命令です。
リスト27ではHLレジスタとBCレジスタを加算し、その結果とDEレジスタを加算し、その結果とSPを加算し、最後にHL＋HLを実行します。途中の結果はそれぞれBC、DE、SPに保存されます。

29－2.　テストプログラムを実行する

　プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　プログラムは一瞬で実行され、最後のHLT命令が繰り返し実行されます。
　最初にBCレジスタの値0123とHLレジスタの値89ABが加算（DAD　B）され、結果の値8ACEはHLレジスタに入れられますが、BCレジスタにもMOV命令で転送されます。
　次にHLレジスタの値8ACEとDEレジスタの値4567とが加算（DAD　D）され、結果の値D035はHLレジスタに入れられますが、DEレジスタにもMOV命令で転送されます。
　次にHLレジスタの値D035とSP（スタックポインタ）の値CDEFとが加算（DAD　SP）され、結果の値9E24はHLレジスタに入れられますが、SPにもSPHL命令で転送されます。
　次にHLレジスタの値9E24が2倍（DAD　H）され、結果の値3C48はHLレジスタに入れられます。
　最後にHLT命令が繰り返し実行されます。
　プログラムが正しく実行されれば、BC＝8ACE、DE＝D035、SP＝9E24、HL＝3C48、になったあと、アドレス0015のHLT命令で停止（繰り返し実行）されます。

　なおDAD命令については、当社ホームページの
「つくるCPU［第212回］」（http://www.alles.or.jp/~thisida/mycpu212.html）～「つくるCPU［第216回］」（http://www.alles.or.jp/~thisida/mycpu216.html）、
でも説明をしています。

30. DAA命令のテスト

　MYCPU80組立説明書　Ⅲ組立　［12］ADD、SUB、CMP、DAD、DAA、RST、INT命令回路　の組立作業後に行う動作テストの説明です。

30－1.　DAA命令のテストプログラムをメモリに書く

　メモリに次のプログラムを書きます（**1.　メモリにデータを書き込む**　の説明を参考にして操作してください）。

```
0000  3E87      MVI  A, 87
0002  C636      ADI  36
0004  4F        MOV  C, A
0005  27        DAA
0006  76        HLT
```
（リスト28）DAA　テストプログラム

　簡単なプログラムです。
　87＋36の計算を行い、その結果に対してDAAを実行します。
　結果を見ればDAAが正しく機能しているかどうかはすぐにわかるのですが、参考までに加算後の（DAA補正前の）値をCレジスタに保存します。

30－2.　テストプログラムを実行する

　プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
　RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

　プログラムは一瞬で実行され、最後のHLT命令が繰り返し実行されます。
　87＋36の加算が行われた結果の値BDがCレジスタに入れられます。
　DAAの結果、Aレジスタには87＋36を十進数で計算したときの答え、123の下位2桁23が入れられ、上位桁への桁上がりを示すキャリーフラグがセットされます。

　なおDAA命令については、当社ホームページの
「つくるCPU［第223回］」（http://www.alles.or.jp/~thisida/mycpu223.html）～「つくるCPU［第228回］」（http://www.alles.or.jp/~thisida/mycpu228.html）、

でも説明をしています。

31. RST命令のテスト

MYCPU80組立説明書 Ⅲ組立 [12]ADD、SUB、CMP、DAD、DAA、RST、INT命令回路 の組立作業後に行う動作テストの説明です。

31－1. 最初に電源を一度OFFにする

今回のテストはレジスタを強制的にクリアしてから始めたいので、最初にこういうことをします。
メモリ書き込みモードの状態で(DS3－4をONにしたままで)、一旦電源をOFFにします。
そうすることによって、レジスタの値をFFにするためです。

電源を一度切って、数秒待ってから再投入します。
すると、プログラムは電池によってバックアップされていますから消えませんが、レジスタの値は消えてしまいます。
電源がONになったとき、レジスタの値はたいていはFFになります(必ずFFになるとは限りません)。
レジスタの状態をクリアしておいてから、メモリにテストプログラムを書きます。

[注記]ここでは先に電源がONになっていて、何かのプログラムがすでに実行されていて、レジスタの値がレジスタごとにばらばらになっていることを想定して、この作業をするように書きましたが、現在電源がOFFの状態で、これからプログラムの書き込みテスト作業をするために電源をONにする場合には、あらためて電源のOFF／ONを繰り返す必要はありません。

31－2. RST命令のテストプログラムをメモリに書く

メモリに次のプログラムを書きます(**1. メモリにデータを書き込む** の説明を参考にして操作してください)。

```
0000  310000    LXI  SP, $0000
0003  04        INR  B
0004  CF        RST  1
0005  76        HLT
0006  00        NOP
0007  00        NOP
0008  3C        INR  A
0009  0C        INR  C
000A  C9        RET
```

(リスト29)RST テストプログラム

RST命令は1バイトのCALL命令です。RST命令はCALLするアドレスが決まっていて、そのアドレスによってRST 0～RST 7の8通りの命令があります。
リスト29ではそのうちのRST 1をテストします。
RST 1はアドレス0008から始まるサブルーチンをCALLします。
サブルーチンをCALLしますから、プログラムの先頭でスタックをセットします。
LXI SPの実行後、INR Bを実行し、RST 1を実行したあとHLTで停止します。
0008から始まるサブルーチンではAレジスタとCレジスタをそれぞれ＋1してからリターンします。
0006、0007にあるNOPに意味はありません。
0005のあと0008までの間には何も無いことを強調するためにNOPを入れました。

31－3. テストプログラムを実行する

プログラムの実行の仕方については **4. テストプログラムを実行する** を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

プログラムは一瞬で実行され、最後のHLT命令が繰り返し実行されます。
Aレジスタ、BレジスタとCレジスタがそれぞれ＋1されています。
RESETSWを押す(押してから離す)と、再びプログラムが実行され、レジスタの値が＋1されて表示されます。
RESETSWを押すたびに(押してから離すごとに)、プログラムが実行されて、レジスタが＋1されます。

リスト29ではAレジスタとCレジスタはRST　1が実行されない限りインクリメントされないので、それが＋1されたということは、RST　1が正しく実行されたことを示しています。

なおRST命令については、当社ホームページの
「つくるCPU[第221回]」(http://www.alles.or.jp/~thisida/mycpu221.html)
でも説明をしています。

32. EI、DI命令(INT回路)のテスト

MYCPU80組立説明書　Ⅲ組立　[12]ADD、SUB、CMP、DAD、DAA、RST、INT命令回路　の組立作業後に行う動作テストの説明です。

32－1.　EI、DI命令(INT回路)のテストプログラムをメモリに書く

メモリに次のプログラムを書きます(**1.　メモリにデータを書き込む**　の説明を参考にして操作してください)。

```
0000  310000    LXI  SP, $0000
0003  AF        XRA  A
0004  FB        EI
0005  00        NOP
0006  00        NOP
0007  04        INR  B
0008  C30700    JMP  $0007

0038  3C        INR  A
0039  FB        EI
003A  C9        RET
```

(リスト30)割込みテストプログラム

リスト30のプログラムは2つの部分に分かれています。
0000～000Aがメインプログラムで0038～003Aが割込みプログラムです。
8080の割込みにはRST命令が利用されます(RST命令については**31.　RST命令のテスト**を参照してください)。
8通りあるRST命令の中でもRST　7(コードFF)は、外部に割込み制御回路を必要とせず、INT割込み信号を与えるだけで割込みを実行させることができるので、8080ではRST　7が割込み命令としてごく普通に利用されます。

MYCPU80はRST　7だけではなくて、RST　0～RST　6命令の割込みも実行できますが、そのためには外部に割り込み制御回路が必要になりますから、ここではその必要のない、RST　7を使って割込みのテストを行います。

プログラムは非常に簡単なものです。
割込みはスタックを利用しますから、スタックポインタの設定が必要です。
プログラムの先頭にLXI　SP命令を置きます。
リセット後は割込みは禁止状態になっていますから、割込みを受け付けるためにはEI命令で割込み許可を与えておく必要があります。
その後はBレジスタをインクリメントし続けます。
0005と0006にNOPがありますが、割込みに必要なものではありません。あとで利用したいことが出てきますからこうしておきます。

RST　7命令を利用した割込みプログラムは0038から書きます。
今回はテストですから割込みプログラムも簡単なものです。
INR　AでAレジスタを＋1するだけです。
割込みが受け付けられると、割込み禁止状態になってしまいますから、メインルーチンに戻る前に割込み許可状態にしておかないと次からの割込みが受け付けられなくなります。
メインルーチンに戻るためのRET命令の直前にEI命令を置きます。
EI命令の次の命令の実行後に割込みが許可されるからです。

32－2.　コネクタケーブルの準備

割込みを実行させるためには外部から割込み信号(INT)を入力する必要があります。

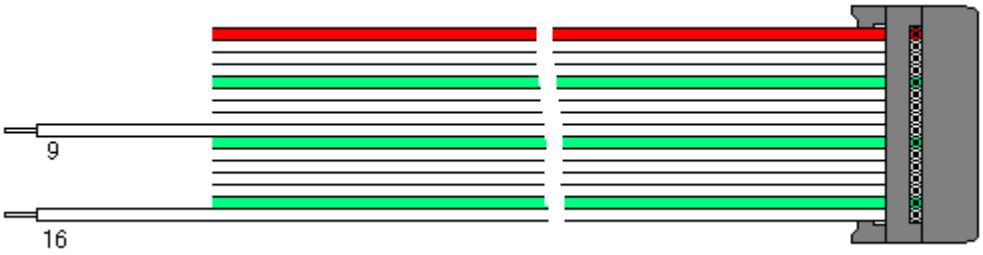
INT信号は16pフラットケーブル用コネクタCN2の9番端子に配線してあります。

CN2端子接続図

RESETIN	1	2	RESETOUT
MEMRD	3	4	MEMWR
IORD	5	6	IOWR
BUSRQ	7	8	BUSAK
INT	9	10	INTRD
EXCLKENL	11	12	EXCLK
T2	13	14	M1
+5V	15	16	GND

なおこのテストでは使いませんが、RST　7以外の割込みを使う場合や、外部にI／O回路を接続するためには、バス信号への接続が必要になります。
それらの信号線は26pフラットケーブル用コネクタCN1に配線してあります（**33.　コネクタ端子接続図**参照）。

16pフラットケーブルのコネクタがついていない方の9番と16番の被覆を先端から1cmほどはがしてください（23－1. テスト用の回路を準備する　を参照してください）。



以上のように作業した16pフラットケーブルコネクタをCN2に接続します。コネクタの△マークと基板面のシルク印刷の△マークを合わせる向きで接続します。このとき16本のコネクタ端子がずれたりしないで全部フラットケーブルコネクタにおさまっていることを確認してください。

32－3.　テストプログラムを実行する

プログラムの実行の仕方については　**4.　テストプログラムを実行する**　を参照してください。
RESETSWを押しながら、ディップスイッチDS3－4をOFFにします。その後RESETSWを離すとプログラムが実行されます。

Bレジスタが高速でインクリメントされるため、BレジスタのLEDは全点灯しているように見えます。
その他のレジスタには変化はありません。
この状態で16pフラットケーブルの9番と16番を一瞬だけショートさせてみます。
少し乱暴な方法ですが今回はテストなので簡単にできる方法でやってみました。
人間の感覚では一瞬に過ぎなくてもコンピュータにとっては非常に長い時間です。
MYCPU80の割込みは、INT入力端子にLレベルの信号を与えることで実行されます。立下りエッジではなくてLの期間中有効な信号として認識されます。
割込みが受け付けられると、アドレス0038からの割り込みプログラムが実行されます。
割込みプログラムの実行が終了してメインプログラムにリターンしてきたときに、まだINT信号がアクティブ（L）になったままだと、再び割込みが発生して、割り込みプログラムが実行されます。

INT入力を一瞬Lにしただけでも、割込み処理に対しては非常に長い時間ですから、数十回以上も割込みが実行されます。
その結果Aレジスタが一瞬でかなりの回数インクリメントされ、LEDにはその結果が表示されます。

上のテストでは16pフラットケーブルの9番と16番を一瞬だけショートさせてみましたが、もっと長い時間ショートさせてみると、どうなるでしょうか。
そうすると、Bレジスタが停止してしまいます。
これは割込み処理が終了してメインプログラムに戻った直後に再び割込みが発生することが繰り返されるため、その間は、メインプログラムの命令が全く実行されなくなるからです。

[注記]MYCPU80はINT端子にLレベルの信号を入力することで割込みが受け付けられますが、CPUによってはHレベルの信号であったり、立下りエッジや立上りエッジであるものなどがあります。
ちなみにZ80はLレベル信号入力ですが8080はHレベル信号入力です。

32－4. DI命令のテスト

リスト30の
0006 00 NOP を
0006 F3 DI に書き換えてください。
前のNOPではなくて後ろのNOPを書き換えます。
念の為、書き換えたあとのメインプログラムのリストを下に示します。

```
0000  310000   LXI  SP, $0000
0003  AF       XRA  A
0004  FB       EI
0005  00       NOP
0006  F3       DI
0007  04       INR  B
0008  C30700   JMP  $0007
```

このようにしてから、32－3. と同じことをしてみます。
すると割込みは受け付けられなくなってしまいます。これはEIの直後にDIが実行されて、それ以後割込みの受け付けが禁止されてしまうからです。
しかし、先に16pフラットケーブルの9番と16番をショートさせたままにしておいて、RESETSWを押す（押してから離す）と、今度は、32－3. と同じように割込み動作が続けられます。
これは先にINT信号がアクティブになっているため、EI命令が実行された直後に割込みが発生し、その後もINT信号がアクティブである間は、割込みプログラムからメインルーチンに戻るとすぐにまた次の割込みが受け付けられてしまうので、DI命令が実行されないまま割込みだけが繰り返し実行されてしまうからです。

一旦16pフラットケーブルの9番と16番を離すと、DI命令が実行されてしまうため、それ以後は9番と16番をショートさせても割込みは発生しなくなります。

ここでEIとDIの間にNOPが入っているのは、割込みが許可されるのはEI命令の次の命令の実行後だからです。
この場合NOP命令の実行後（DI命令の直前）に割込みが可能になります。
もしもNOPがないと、割込みが可能になるのはDI命令の実行後になるため、実際には割込み可能にはならなくなってしまいます。
そのことを確認するために、0005にF3を書き、0006に00を書いてから、同じようにテストをしてみてください。

なお割込みについては、当社ホームページの
「つくるCPU［第242回］」（http://www.alles.or.jp/~thisida/mycpu242.html）～「つくるCPU［第246回］」
（http://www.alles.or.jp/~thisida/mycpu246.html）
でも説明をしています。

33. コネクタ端子接続図

CN1

D0	1	2	D1
D2	3	4	D3
D4	5	6	D5
D6	7	8	D7
A0	9	10	A1
A2	11	12	A3
A4	13	14	A5
A6	15	16	A7
A8	17	18	A9
A10	19	20	A11
A12	21	22	A13
A14	23	24	A15
＋5V	25	26	GND

CN2

RESETIN	1	2	RESETOUT
MEMRD	3	4	MEMWR
IORD	5	6	IOWR
BUSRQ	7	8	BUSAK
INT	9	10	INTRD
EXCLKENL	11	12	EXCLK
T2	13	14	M1
+5V	15	16	GND

CN3

IN7	1	2	IN6
IN5	3	4	IN4
IN3	5	6	OUT7
OUT6	7	8	OUT5
＋5V	9	10	GND