THE COMPUTER

THE PAGES WHICH FOLLOW ARE AN INTRODUCTION TO COMPUTERS. OUR AIM IS TO BREAK DOWN THE MYSTERY OF BITS, BYTES, PROCESSORS AND ALL THOSE TERMS RELATING TO THE 'MICRO' — INTO UNDERSTANDABLE FORM.

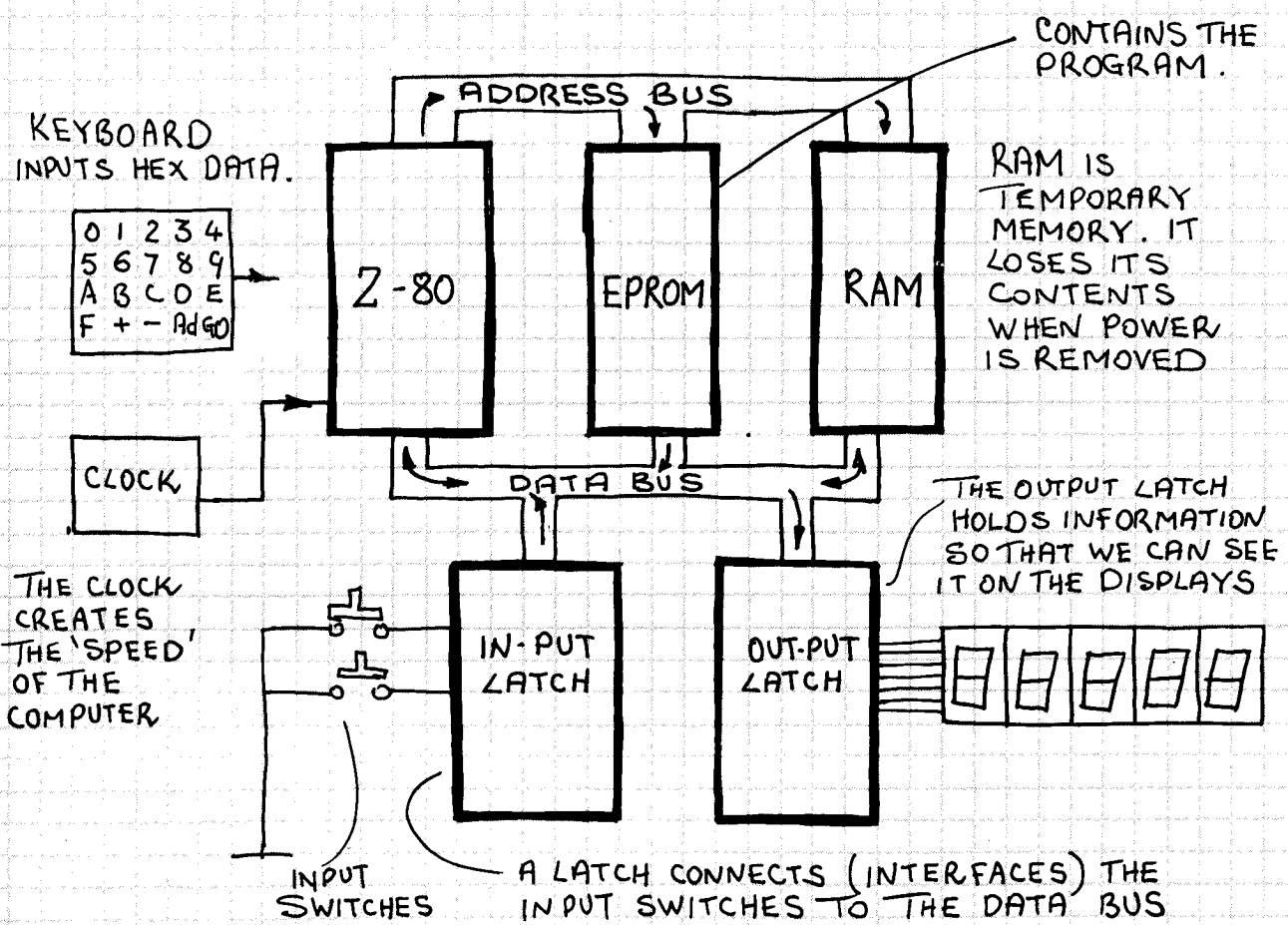
THE MICRO IS HERE TO STAY — SO WE HAVE TO LIVE WITH IT. AS ELECTRONICS ENTHUSIASTS, THIS WILL BE TO OUR ADVANTAGE AS IT OFFERS MORE SCOPE FOR DESIGNING THAN USING INDIVIDUAL CHIPS & HAS MUCH MORE FLEXIBILITY BECAUSE A MICRO SYSTEM PERFORMS ITS OPERATIONS VIA A PROGRAM.

CHANGE THE PROGRAM & YOU CHANGE THE OUTCOME.

MICRO'S COMBINE THE SKILL OF ELECTRONICS WITH THE ART OF PROGRAMMING & REQUIRES A NEW APPROACH TO PROJECT DESIGN.

THAT'S WHY WE NEED TO START AT THE BEGINNING.

HERE ARE SOME TERMS & A BLOCK DIAGRAM TO ACQUAINT YOU WITH THE 'MYSTERY' WE CALL THE MICRO.



THE ARROWS SHOW THE DIRECTION OF FLOW OF DATA.

WHAT A COMPUTER WILL DO....

A COMPUTER IS SIMILAR TO A BOOK. THE PAGES REPRESENT THE CHIPS & THE STORY IS EQUIVALENT TO THE PROGRAM.

JUST AS YOU CAN WRITE AN ENDLESS NUMBER OF STORIES, YOU CAN CREATE A PROGRAM TO SUIT ALMOST ANY SITUATION.

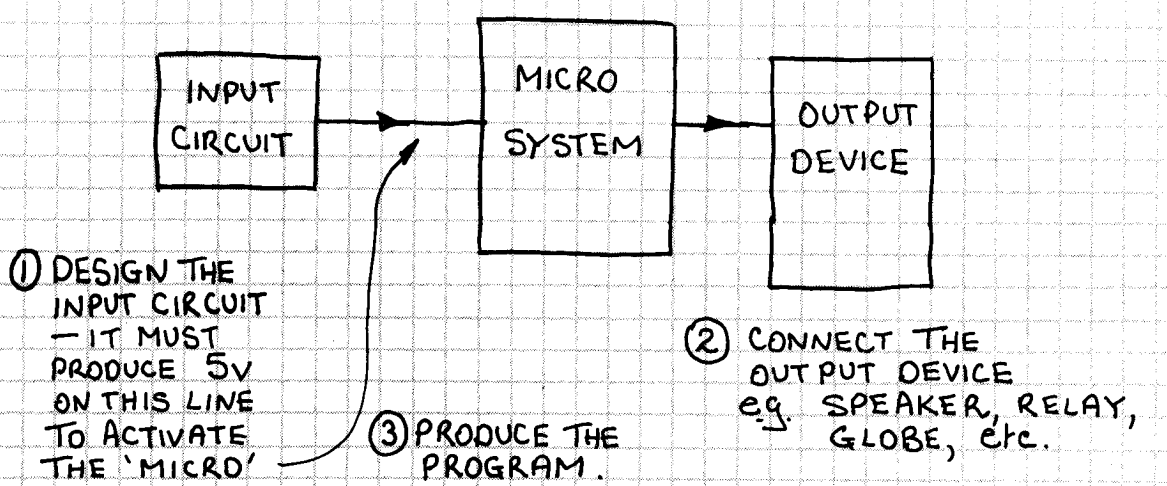
PROGRAMS ARE CONTAINED IN AN EPROM & THE LEVEL OF PERFORMANCE DEPENDS ON THE SKILL OF THE PROGRAMMER.

PROGRAMS CAN RANGE FROM A SIMPLE SEQUENCE TO STRATEGY CAPABLE OF PLAYING CHESS — AND WINNING!

IN FACT A COMPUTER WILL DO ALMOST ANYTHING. ANY PROJECT USING ABOUT 8 OR MORE IC'S CAN BE CONVERTED TO A MICRO DESIGN.

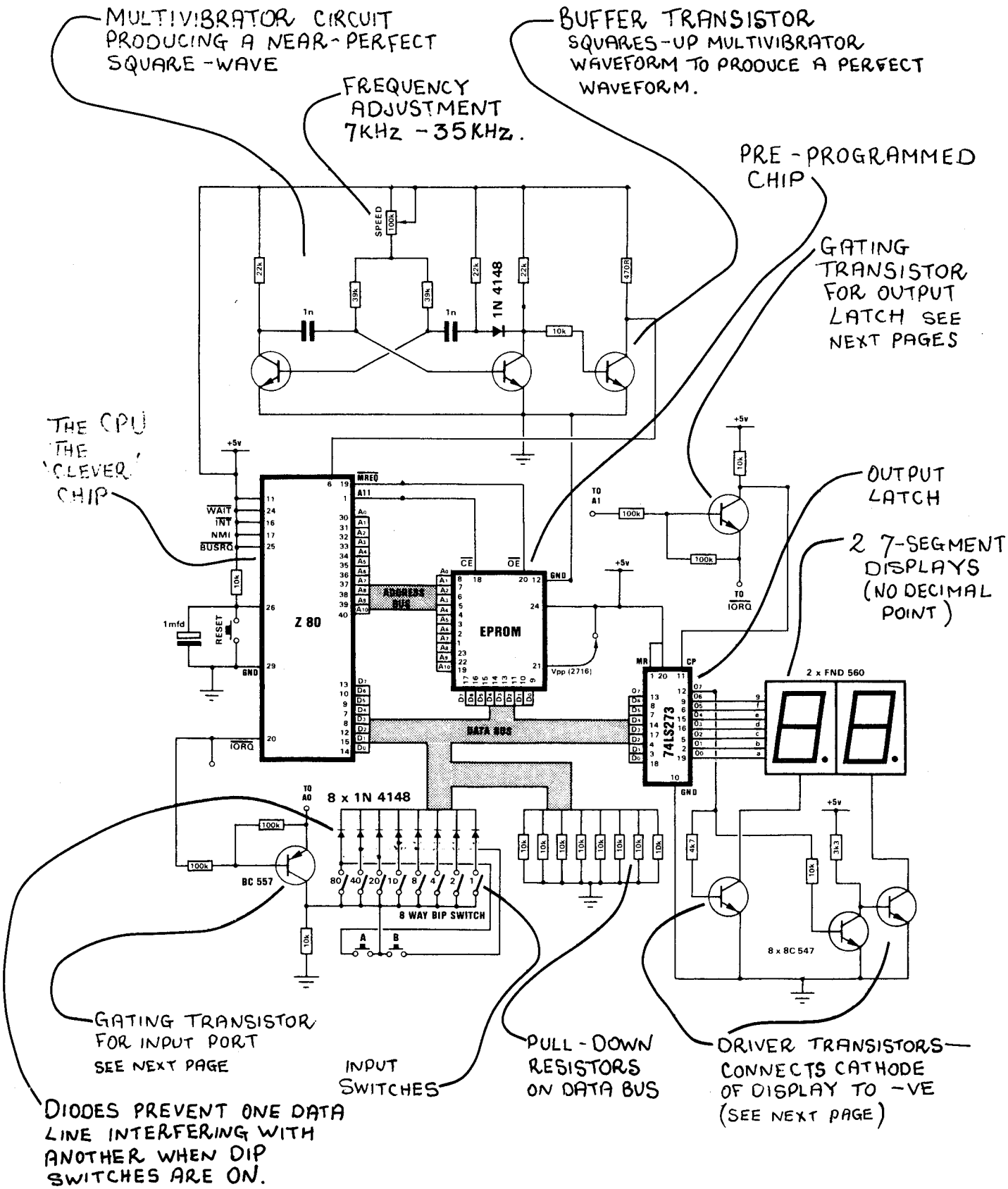
WE INTEND TO EXPLAIN BOTH THE SOFTWARE & HARDWARE SIDE OF CREATING A MICRO PROJECT & SHOW HOW TO GET IT ALL TOGETHER.

HERE ARE THE 3 STEPS:



YOU WILL GAIN A LOT BY READING THE ARTICLE ON THE TALKING ELECTRONICS COMPUTER 'TEC-1A' IN TE MAGAZINE, ISSUES 10 →, & THE MICROCOMP PROJECT IN ISSUES 13 →.

EVEN MORE WILL BE GAINED BY BUILDING ONE OR BOTH MODELS AS ALL OUR DISCUSSION WILL BE CENTRED AROUND THEM.


MULTIVIBRATOR CIRCUIT PRODUCING A NEAR-PERFECT SQUARE-WAVE
BUFFER TRANSISTOR SQUARES-UP MULTIVIBRATOR WAVEFORM TO PRODUCE A PERFECT WAVEFORM.
FREQUENCY ADJUSTMENT 7KHz - 35KHz.
PRE-PROGRAMMED CHIP
GATING TRANSISTOR FOR OUTPUT LATCH SEE NEXT PAGES
THE CPU 'THE CLEVER' CHIP
OUTPUT LATCH
2 7-SEGMENT DISPLAYS (NO DECIMAL POINT)
SPEED
100k
22k
39k
1n
1N 4148
10k
470R
+5v
WAIT
INT
NMI
BUSRQ
RESET
1mfd
GND
Z 80
MREQ
A11
ADDRESS BUS
EPROM
CE
OE
Vpp (2716)
DATA BUS
IORQ
TO A1
TO IORQ
MR
CP
74LS273
2 x FND 560
TO A0
8 x 1N 4148
BC 557
8 WAY DIP SWITCH
4k7
3k3
8 x BC 547
GATING TRANSISTOR FOR INPUT PORT SEE NEXT PAGE
INPUT SWITCHES
PULL-DOWN RESISTORS ON DATA BUS
DRIVER TRANSISTORS CONNECTS CATHODE OF DISPLAY TO -VE (SEE NEXT PAGE)
DIODES PREVENT ONE DATA LINE INTERFERING WITH ANOTHER WHEN DIP SWITCHES ARE ON.

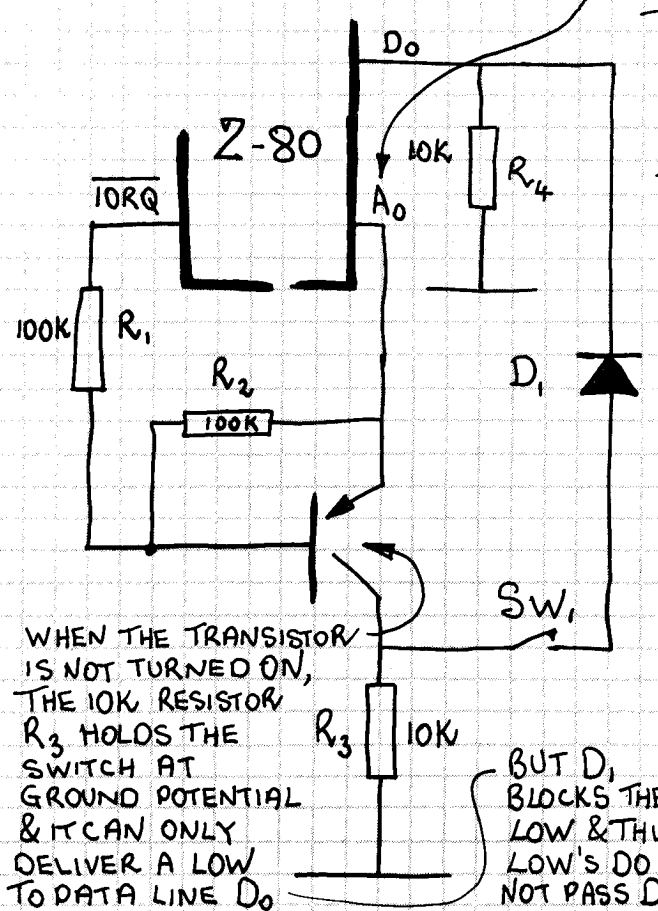
THE INPUT GATING TRANSISTOR

THE IN-OUT REQUEST LINE ($\overline{IORQ}$) OF THE Z-80 IS A SOFTWARE DRIVEN OUTPUT PIN. THIS MEANS IT IS ACTIVATED (GOES LOW) AS A RESULT OF A STATEMENT IN A PROGRAM, SUCH AS IN A(01), OUT (01) A [OR (02) (04) (08) ETC].

TO GET IT TO SELECT EITHER THE 'IN' FUNCTION OR 'OUT' FUNCTION WE MUST COMBINE $\overline{IORQ}$ WITH ANOTHER LINE.

IN A SIMPLE CIRCUIT SUCH AS WE ARE DESCRIBING, WE HAVE CHOSEN THE FIRST ADDRESS LINE (A_0) FOR THE INPUT PORT & A_1 FOR THE OUTPUT PORT. THIS PRODUCES: IN A(01) OUT (02) A.

HOW THE CIRCUIT WORKS



THE VOLTAGE (AND CURRENT) TO DRIVE THE CIRCUIT COMES FROM LINE A_0

WHEN $\overline{IORQ}$ IS LOW AND A_0 IS HIGH THE TRANSISTOR IS TURNED ON. THE VOLTAGE FOR DATA LINE D_0 COMES FROM A_0, VIA THE TRANSISTOR, SWITCH & DIODE TO D_0.

R_1 SEPARATES $\overline{IORQ}$ FROM THE BASE.

R_2 PREVENTS THE TRANSISTOR TURNING ON WHEN NOT BEING ACCESSED.

R_3 KEEPS THE COLLECTOR AT ZERO WHEN THE TRANSISTOR IS NOT TURNED ON.

R_4 PREVENTS THE DATA LINE FLOATING WHEN NO DATA IS PRESENT.

D_1 PREVENTS ONE DATA LINE UPSETTING THE OTHERS WHEN A SWITCH IS PRESSED & THE TRANSISTOR IS OFF.

WHEN THE TRANSISTOR IS NOT TURNED ON, THE 10K RESISTOR R_3 HOLDS THE SWITCH AT GROUND POTENTIAL & IT CAN ONLY DELIVER A LOW TO DATA LINE D_0

BUT D_1 BLOCKS THE LOW & THUS LOW'S DO NOT PASS D_1

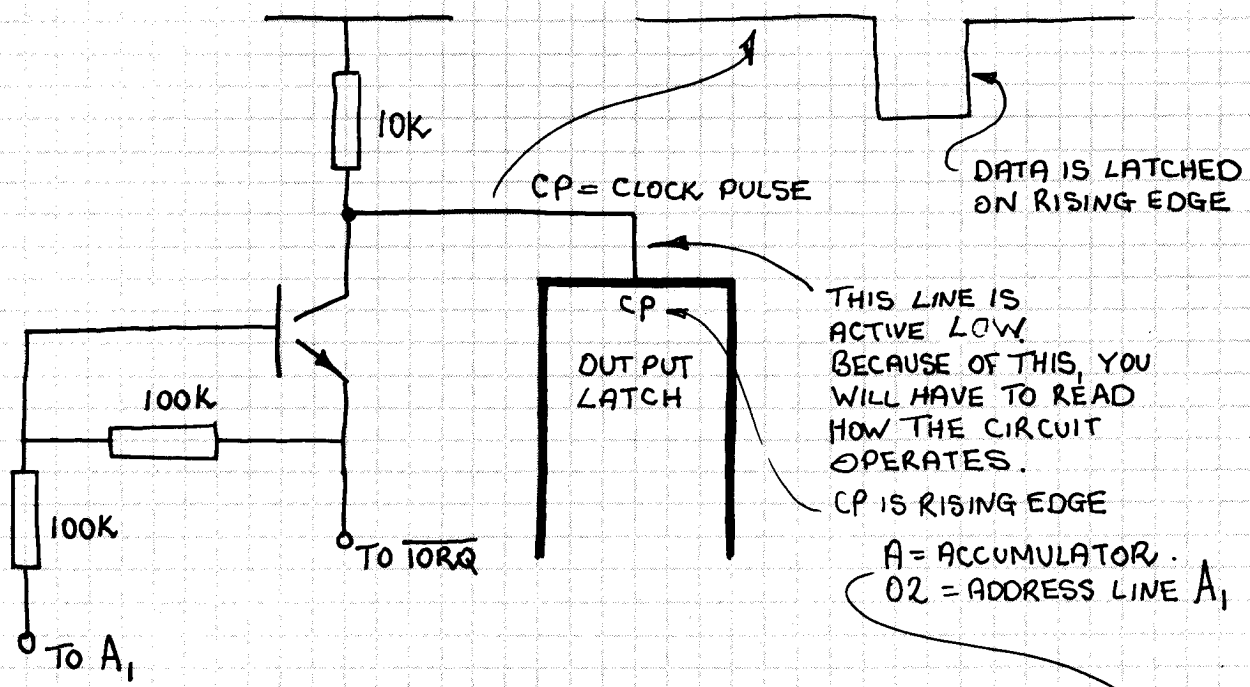
THIS CIRCUIT ONLY COMES INTO OPERATION WHEN A STATEMENT SUCH AS IN A(01) IS BEING EXECUTED. IF THE SWITCH IS PRESSED, THE RESULT WILL BE A HIGH ON D_0, IF NOT PRESSED, A LOW WILL BE REGISTERED.

THIS IS THE ONLY TIME WHEN THE SWITCH IS DETECTED. AT ALL OTHER TIMES A CLOSED SWITCH WILL PLACE A LOW ON THE DATA BUS OR MORE ACCURATELY – IT WILL PLACE NIL ON THE DATA BUS DUE TO DIODE D_1.
THIS CAPABILITY (OF THE CIRCUIT) IS CALLED TRI-STATE. (3 STATES)

THE OUTPUT GATING TRANSISTOR

THE OBJECT OF THE CIRCUIT IS TO GATE (COMBINE) A_1 & $\overline{IORQ}$ TO ACTIVATE THE OUTPUT LATCH.

THE CIRCUIT SITS WITH CP LINE HIGH. IT PULSES LOW & ON THE RISING EDGE OF THE PULSE, DATA IS LATCHED INTO THE CHIP.



THIS IS THE SEQUENCE TO LATCH DATA INTO THE OUTPUT LATCH.

1. THE Z-80 RECEIVES AN OUTPUT STATEMENT: OUT (02) A.
2. THIS CAUSES THE DATA IN THE ACCUMULATOR TO APPEAR ON THE DATA BUS & ADDRESS LINE A_1 GOES HIGH.
3. A SHORT TIME LATER THE $\overline{IORQ}$ LINE GOES LOW.
4. THIS TURNS THE TRANSISTOR ON, PULLING THE CP LINE OF THE LATCH LOW.
   THE DATA DOES NOT ENTER THE LATCH YET.
5. THE $\overline{IORQ}$ LINE GOES HIGH LATCHING DATA INTO THE LATCH & ON A DISPLAY ETC.
6. THE Z-80 ADVANCES TO THE NEXT STATEMENT IN THE PROGRAM.
7. THE DISPLAY REMAINS ILLUMINATED.

A SAMPLE PROGRAM:

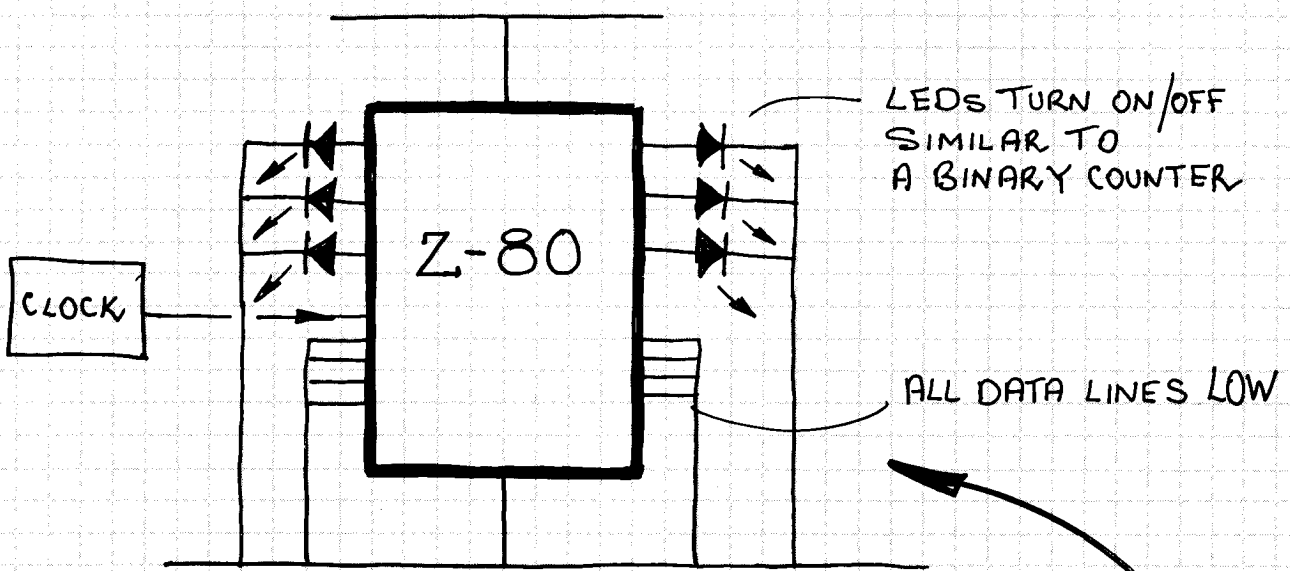
```
LD A 00
OUT (02) A   - LATCH OUTPUTS 00.
LD A 04      "      "      "
OUT (02) A   - LATCH OUTPUTS 04
LD A 0B      "      "      "
OUT (02) A   - LATCH OUTPUTS 0B
etc.         "      "      "
             "      "      "
```

THE LATCH CHANGES FROM OUTPUT 00 TO 04 ONLY AFTER THE 'OUT' INSTRUCTION HAS BEEN EXECUTED.

THE Z-80 CPU — HOW IT WORKS

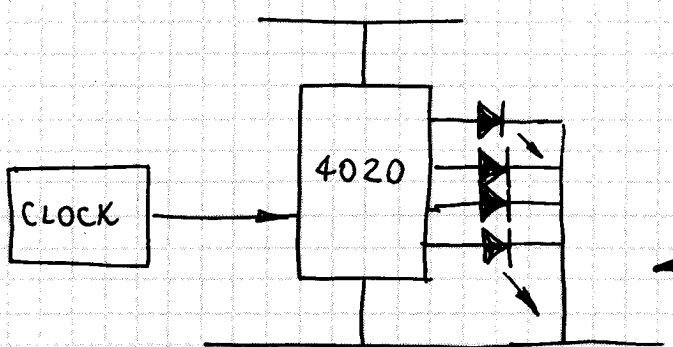
THE Z-80 AS A BINARY DIVIDER!!

THE ADDRESS LINES ON A Z-80 TURN ON-OFF VERY SIMILAR TO A BINARY DIVIDER CHIP WHEN CONNECTED AS FOLLOWS:



WHEN ALL DATA LINES ARE LOW THE Z-80 EXECUTES NO-OPERATIONS — A COMMAND WHICH INCREMENTS THE VALUE ON THE ADDRESS LINES.

LEDS ON THE ADDRESS LINES TURN ON/OFF IN A SIMILAR MANNER TO THE OUTPUTS OF A BINARY COUNTER, SUCH AS A 4024, 4020 OR 4040.



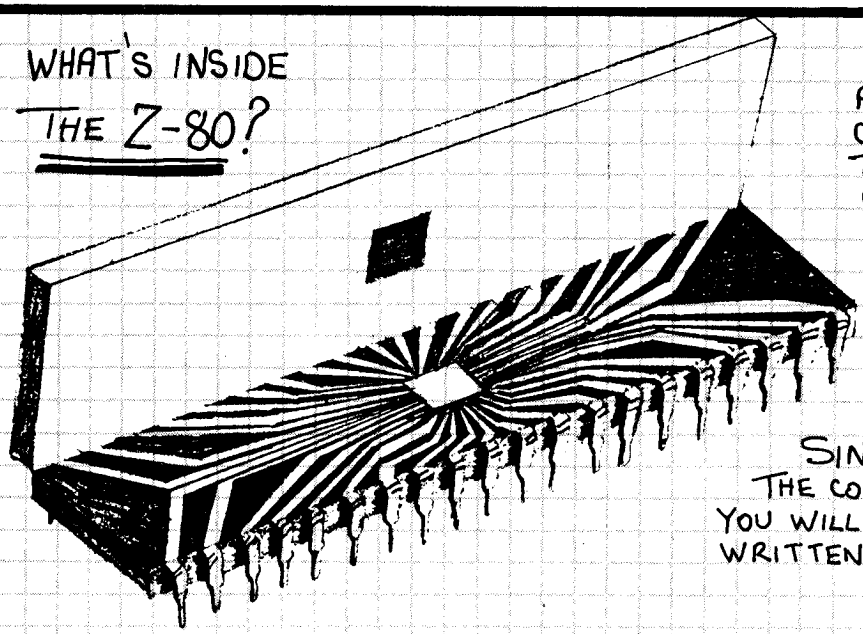
A BINARY COUNTER

BOTH OPERATE VERY SIMILAR WHEN WIRED LIKE THIS!

SUMMARY: DATA VALUE 00 CREATES NOP'S — NO-OPERATIONS IN WHICH THE Z-80 MERELY INCREMENTS THE VALUE OF THE ADDRESS BUS.

[SOME Z-80 LINES PULSE — WE ARE KEEPING THE DISCUSSION SIMPLE]

WHAT'S INSIDE THE Z-80?



ALL THE 'WORKS' ARE CONCENTRATED IN A TINY CHIP IN THE CENTRE OF THE IC.

THE SIZE OF THE IC IS DICTATED BY THE NUMBER OF PINS & NOT THE COMPLEXITY OF THE ELECTRONICS.

SINCE YOU CAN'T SEE THE COMPONENTS IN THE Z-80 YOU WILL HAVE TO ACCEPT WRITTEN EXPLANATIONS.

BASICALLY THE Z-80 CONSISTS OF 'A COMPUTER INSIDE A CHIP'. IT HAS IS OWN PROGRAMMED MEMORY, BUS LINES & SET OF COMMANDS WHICH IT RECOGNISES.

WE ARE GOING TO DEAL WITH ONE SMALL PORTION – CALLED THE SET OF REGISTERS. THESE ARE THE MAIN ITEMS WE ARE CONCERNED WITH. WHEN WE WRITE A PROGRAM FOR THE Z-80 WE ARE WRITING INSTRUCTIONS FOR THE <u>REGISTERS</u>.

A REGISTER IS SIMPLY A GROUP OF 8 FLIP-FLOPS. IT CAN BE DRAWN AS A SET OF 8 BOXES:

EACH CELL ACCEPTS A BINARY VALUE WHICH CAN ONLY BE 0 OR 1.

e.g.

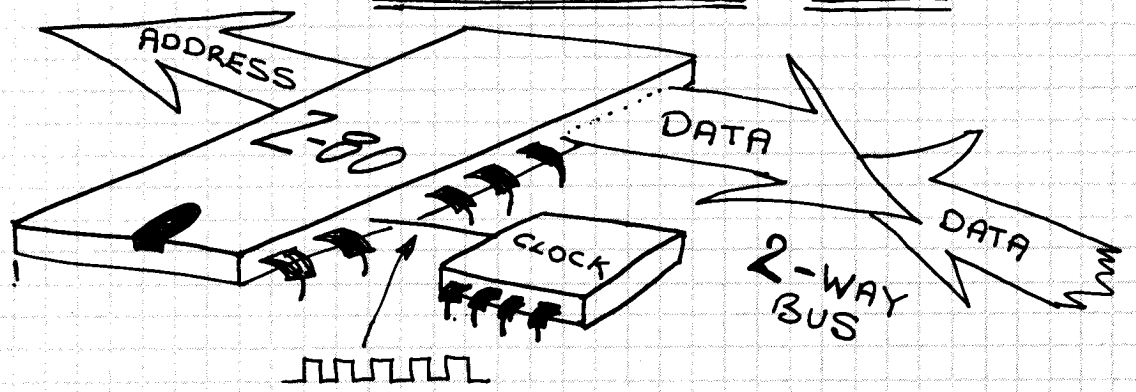
WHEN A CELL CONTAINS A '1' IT IS <u>SET</u>. WHEN IT CONTAINS A '0' IT IS <u>RESET</u>.

THE VALUE OF A CELL IS CALLED A BIT AND THESE ARE NUMBERED AS FOLLOWS:

IN THE EXAMPLE ABOVE: bit 0 = 1; bit 1 = 0, bit 2 = 0; bit 3 = 1 bit 4 = 1 bit 5 = 0; bit 6 = 0 & bit 7 = 1.

THERE ARE A NUMBER OF THINGS WE CAN DO TO THIS REGISTER & THIS IS WHAT MAKES THE MICROPROCESSOR SO DIFFERENT TO STORING BITS IN INDIVIDUAL 'FLIP-FLOP' CHIPS.

THE Z-80 CPU



— THE Z-80 HAS 16 ADDRESS LINES & 8 DATA LINES
THE ARE CALLED THE ADDRESS BUS & DATA BUS.

— INFORMATION ENTERS THE Z-80 VIA THE DATA BUS & EMERGES VIA THIS BUS — IT IS A TWO-WAY BUS.

— THE ADDRESS BUS IS A ONE-WAY BUS. THE Z-80 SENDS OUT ADDRESSES VIA THIS BUS TO MEMORY

— MEMORY CAN BE EITHER RAM OR ROM.
RAM IS TEMPORY STORAGE MEMORY
ROM IS PERMANENT STORAGE MEMORY.

— ALL INFORMATION IS IN THE FORM OF HIGHS & LOWS.

— DATA RANGES FROM 00000000 TO 11111111.
THIS GIVES 256 DIFFERENT COMBINATIONS.

— THE Z-80 OPERATES ON BINARY 'BITS'.
A 'BIT' IS A DIGIT WHICH IS EITHER HIGH OR LOW.
4 BITS = ONE NIBBLE
8 BITS = ONE BYTE
] THIS REFERS TO THE DATA BUS

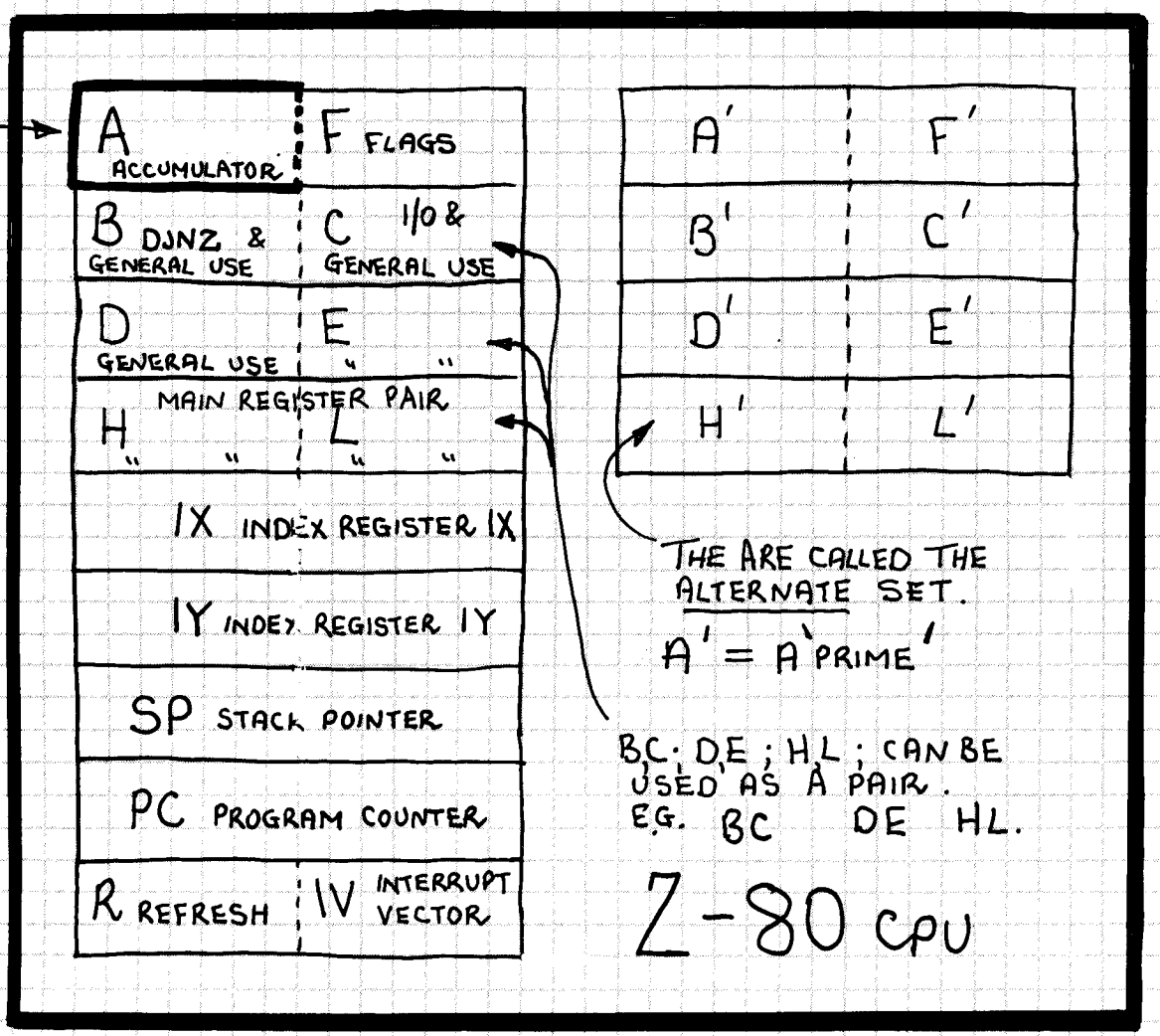
— THE FIRST REQUEST SENT OUT BY A Z-80 ON START-UP IS SENT VIA ADDRESS 00000000.
THE FIRST PIECE OF DATA IT RECEIVES WILL BE INTERPRETED AS AN INSTRUCTION. DEPENDING ON THE FIRST INSTRUCTION, THE Z-80 MAY REQUEST ANOTHER ONE, TWO OR THREE BYTES TO COMPLETE THE COMMAND.

— INFORMATION RECEIVED ON THE DATA BUS WILL BE INTERPRETED AS AN INSTRUCTION, DATA-VALUE OR JUMP VALUE etc DEPENDING ON WHERE IT IS, IN THE PROGRAM.

— THE Z-80 WILL OPERATE ON A CLOCK FREQUENCY AS LOW AS 7KHz — UP TO A MAXIMUM OF 2·5 MHz. THE Z-80A WILL OPERATE UP TO 4 MHz.

— THE LETTERS 'CPU' STAND FOR CENTRAL PROCESSING UNIT.

THE CPU REGISTER SET: FOR THE Z-80



MOST OF THE OPERATIONS ARE CARRIED OUT IN THE ACCUMULATOR. THIS IS BECAUSE THE ACCUMULATOR HAS A LOT OF FEATURES WHICH THE OTHER REGISTERS DO NOT HAVE.

ONCE AN OPERATION IS COMPLETE, THE RESULT(S) ARE TRANSFERRED TO MEMORY (RAM) OR TO OTHER REGISTERS.

THE B REGISTER HAS A SPECIAL DJNZ FEATURE IN WHICH IT IS AUTOMATICALLY DECREMENTED BY ONE EACH TIME A LOOP IS PERFORMED. THE PROGRAM ADVANCES WHEN B IS ZERO.

REGISTERS B & C CAN BE USED SEPARATELY OR AS A PAIR. THE SAME APPLIES TO D & E AND H, L.

REGISTER PAIR HL CAN BE REMEMBERED AS HIGH, LOW. H IS THE HIGH BYTE IN A PROGRAM AND L IS THE LOW BYTE. THIS ALSO APPLIES TO B, C. B IS THE HIGH BYTE & C IS THE LOW BYTE. ALSO D IS THE HIGH BYTE & E IS THE LOW BYTE.

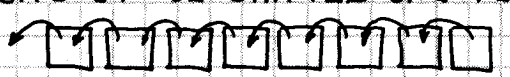
THE ACCUMULATOR

MOST OF THE OPERATIONS IN A MICROPROCESSOR ARE CARRIED OUT IN THE ACCUMULATOR. THIS CONSISTS OF A SET OF 8 FLIP-FLOPS WITH A LOT OF SUPPORT CIRCUITRY, SO THAT THE FOLLOWING CAN BE PERFORMED: (OTHER REGISTERS CAN DO SOME OF THE FOLLOWING, BUT NOT ALL)

— ANY BIT CAN BE SET TO 1 OR RESET TO ZERO.

— ALL BITS CAN BE SET TO ZERO

— ALL BITS CAN BE SHIFTED ONE PLACE TO THE LEFT:



— ALL BITS CAN BE SHIFTED ONE PLACE TO THE RIGHT:

— ANY BIT CAN BE TESTED TO SEE IF IT IS A '1' OR '0'.

— THE VALUE OF THE REGISTER CAN BE INCREMENTED BY ONE.

— " " " " " " " DECREMENTED " "

— EACH '0' CAN BE CHANGED INTO A '1' & EACH '1' INTO A '0'.

— A VALUE CAN BE ADDED TO THE REGISTER

— A VALUE CAN BE SUBTRACTED FROM THE REGISTER

— AN 8 BIT BINARY VALUE CAN BE CONVERTED TO TWO 4-BIT BCD VALUES.

— A NUMBER OF CONDITIONAL CALLS OR JUMPS CAN BE PERFORMED ONLY WHEN A CERTAIN CONDITION IS MET.
FOR EXAMPLE: CALL NZ, ADDRESS — THE CALL WILL ONLY BE PERFORMED IF THE RESULT OF THE PREVIOUS INSTRUCTION IS NOT ZERO.

— THE ACCUMULATOR CAN BE COMPARED TO A VALUE OF DATA, TO OTHER REGISTERS, OR TO A DATA VALUE IN MEMORY.

— THE ACCUMULATOR CAN BE SAVED BY AN OPERATION CALLED PUSH. THIS PLACES THE VALUE INTO A PRE-DETERMINED AREA OF MEMORY CALLED THE STACK.

— THE ACCUMULATOR CAN ALSO PERFORM LOGIC FUNCTIONS: AND, OR, NAND, NOR, EX-OR, EX-NOR.

THIS FORMS THE BASIS TO ALL PROGRAMS & WE WILL SHOW HOW EACH SENTENCE IS CONVERTED TO A MACHINE CODE INSTRUCTION WHICH THE PROCESSOR UNDERSTANDS.

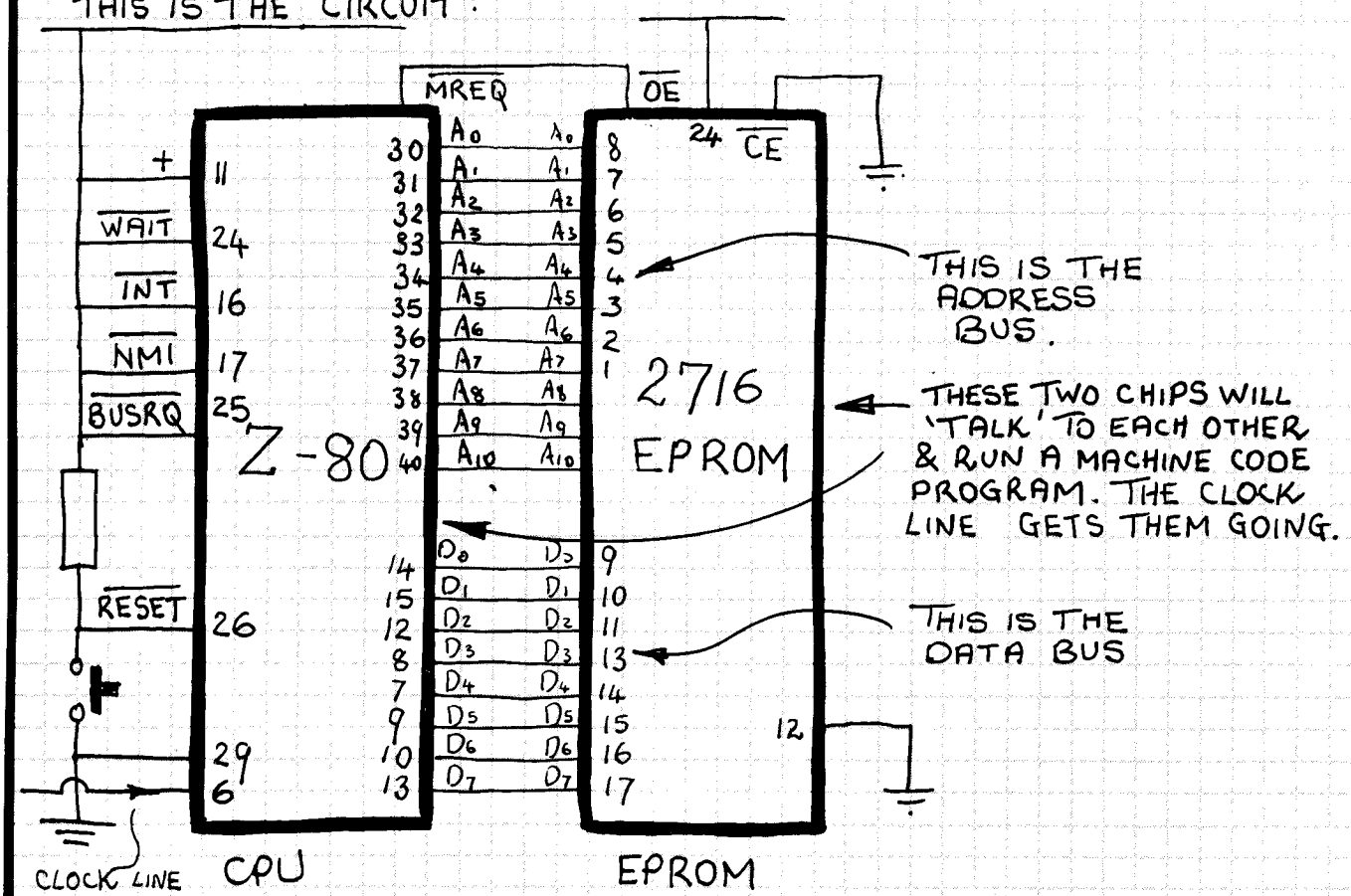
THE Z-80 HOW IT WORKS

CONNECTING THE Z-80 TO AN EPROM.

WHEN THE Z-80 IS CONNECTED TO A CHIP CONTAINING A PROGRAM, SUCH AS AN EPROM, THE TWO WILL BE CAPABLE OF TALKING TO EACH OTHER AND THE PROGRAM WILL BE EXECUTED.

THIS MEANS THE OUTPUT (RESULT) OF THE PROGRAM WILL NOT BE AVAILABLE FOR US TO SEE. BUT AT THIS STAGE WE WANT TO SEE HOW THE TWO ARE CONNECTED TOGETHER.

THIS IS THE CIRCUIT:



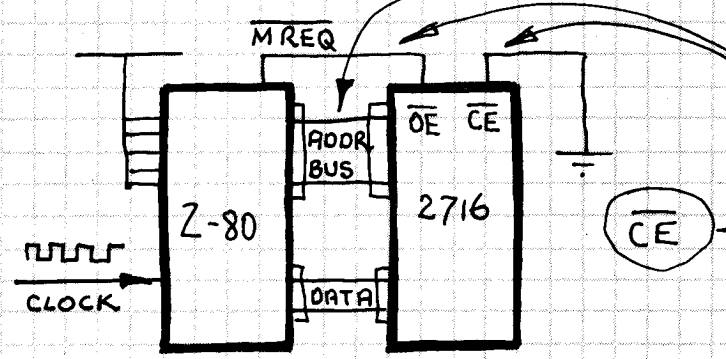
TWO BUSES ARE REQUIRED BETWEEN Z-80 & EPROM.— AN ADDRESS BUS & A DATA BUS.

IN COMPUTERS, ALL LINE NUMBERING STARTS AT ZERO. THUS WE HAVE 'A_0 & D_0 AS THE FIRST LINES. THIS IS IN LINE WITH COUNTING, WHICH STARTS AT ZERO.

THE ADDRESS BUS (IN THE CONNECTION ABOVE) HAS 11 LINES & 8 LINES IN THE DATA BUS. 11 ADDRESS LINES WILL ADDRESS FROM 0000 TO 07FF. THIS IS 2K OF MEMORY AS WILL BE EXPLAINED LATER. [THE Z-80 HAS 16 ADDRESS LINES)

THE 2716 EPROM

THE DIAGRAM ON THE PREVIOUS PAGE CAN BE SIMPLIFIED BY DRAWING THE BUSES AS A PATHWAY:

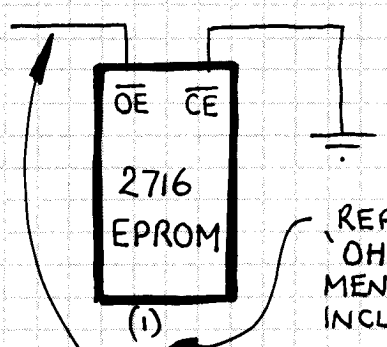


THESE ARE THE LINES WE ARE DISCUSSING.

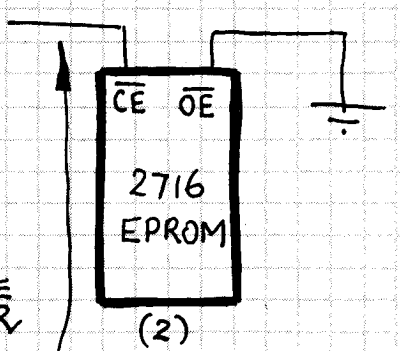
$\overline{CE}$ — THIS SYMBOL MEANS THE CHIP WILL BE 'TURNED ON' WHEN THE LINE IS LOW. IT IS READ AS: SEE-EEE-BAR

THE EPROM MUST BE TURNED ON ONLY DURING THE TIME WHEN IT IS REQUIRED. THIS IS BECAUSE THE DATA & ADDRESS BUSES ARE REQUIRED BY OTHER CHIPS IN THE SYSTEM.

THIS MEANS ONE OR MORE LINES ARE NEEDED BETWEEN THE Z-80 & EPROM. THERE ARE TWO CONTROL LINES AND THEY ARE CALLED CHIP ENABLE & OUTPUT ENABLE. WHEN BOTH OF THESE LINES GO LOW THE VALUE ON THE ADDRESS BUS WILL CAUSE DATA TO BE OUTPUTTED BY THE EPROM ON THE DATA BUS.



READ THIS AS: 'OH-EEE' AND MENTALLY NOTE THE INCLUSION OF THE BAR



TAKING $\overline{OE}$ HIGH CAUSES THE OUTPUT LINES OF THE EPROM TO GO INTO A HIGH IMPEDANCE STATE. OUTPUT DATA IS SUPPLIED VERY QUICKLY AFTER $\overline{OE}$ IS TAKEN LOW.

TAKING $\overline{CE}$ HIGH SEE(2) CAUSES THE OUTPUT LINES TO GO INTO A HIGH IMPEDANCE STATE AND ALSO A 'POWER-DOWN' STATE. BUT WHEN $\overline{CE}$ GOES LOW, THE EPROM TAKES CONSIDERABLY LONGER TO PROVIDE DATA ON THE OUTPUT PINS.

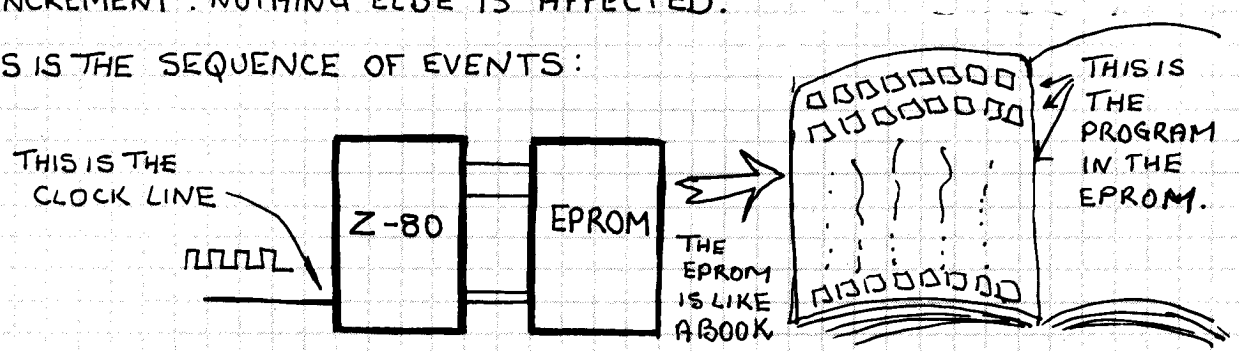
A HIGH IMPEDANCE STATE IS IDENTICAL TO SAYING 'TRI-STATE'; SO THAT THEY PUT NO LOAD ON THE BUSES.

THE Z-80 - HOW IT WORKS

LET US LOOK AT THE EXECUTION OF A SINGLE BYTE INSTRUCTION.— CALLED NO OPERATION.

THIS INSTRUCTION CAUSES THE PROGRAM COUNTER (INSIDE THE Z-80) TO INCREMENT. NOTHING ELSE IS AFFECTED.

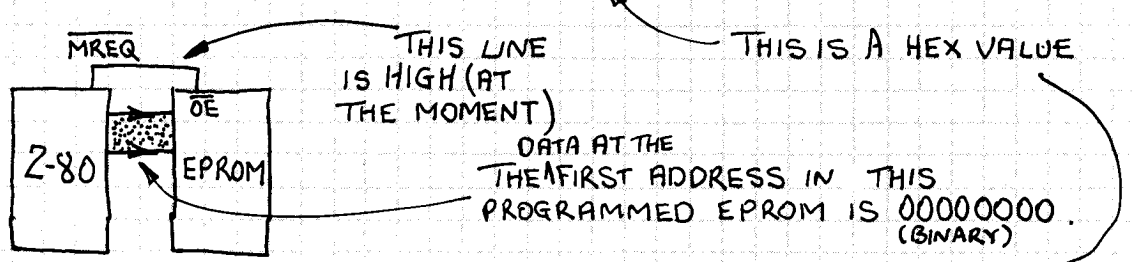
THIS IS THE SEQUENCE OF EVENTS:



AN INSTRUCTION IS NOT CARRIED OUT DURING EACH CLOCK CYCLE. IT REQUIRES A NUMBER OF CLOCK CYCLES TO PERFORM AN INSTRUCTION. (BETWEEN 4 & 23) BECAUSE THE Z-80 HAS INTERNAL HOUSE KEEPING OPERATIONS TO PERFORM SO THAT THE INSTRUCTION CAN BE CARRIED OUT.

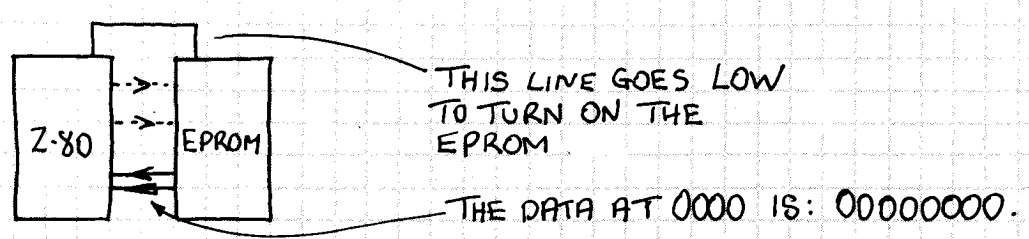
IT TAKES 4 CLOCK CYCLES TO PERFORM A NO OPERATION INSTRUCTION.

① THE VALUE IN THE PROGRAM COUNTER (0000) IS OUTPUTTED TO THE ADDRESS BUS:



② $\overline{\text{MREQ}}$ LINE GOES LOW & TURNS ON THE EPROM.

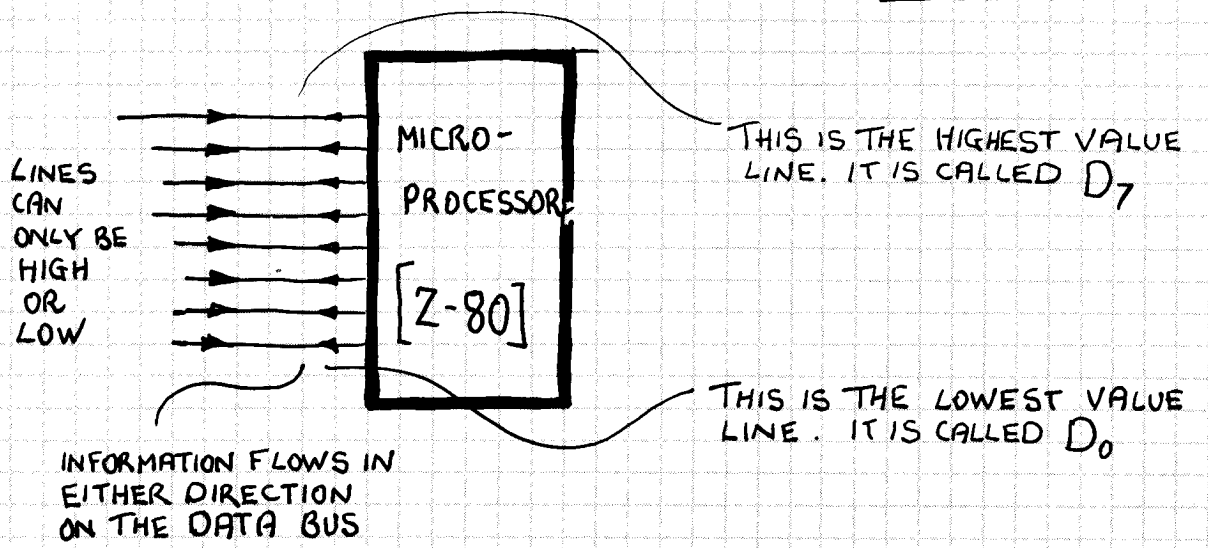
③ THE EPROM RESPONDS BY SENDING THE DATA LOCATED AT 0000 TO THE Z-80:



④ THE Z-80 INTERPRETS 00000000 AS A SINGLE BYTE INSTRUCTION & CARRIES OUT THE TASK OF INCREASING (INCREMENTING) THE PROGRAM COUNTER TO ···001 (HEX) (0000000000000001 BINARY) THE MREQ LINE GOES HIGH IMMEDIATELY THE COMPUTER HAS FINISHED READING THE VALUE.

BITS, NIBBLES & BYTES

IN AN 8-BIT MICROPROCESSOR SYSTEM (SUCH AS Z-80) 8 LINES OF INFORMATION IS PRESENTED TO THE PROCESSOR AT THE SAME TIME.

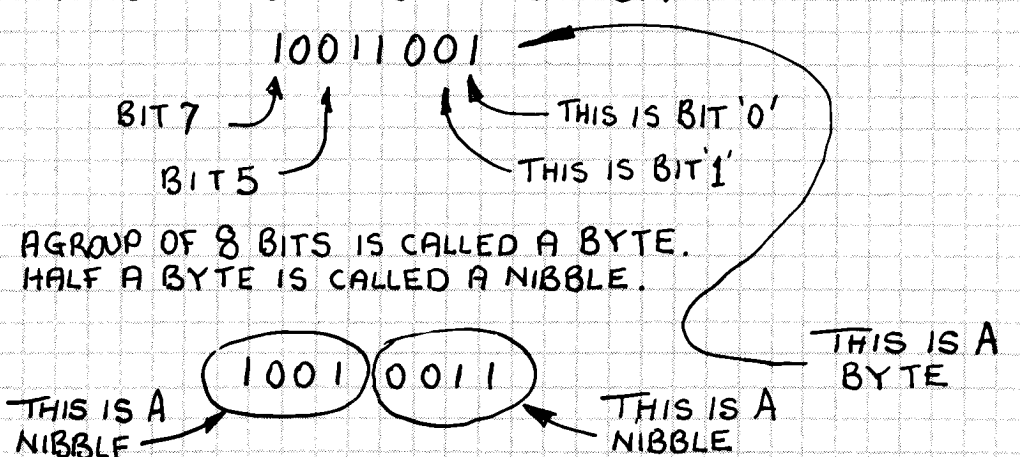


WE DESCRIBE THE POWER OF A PROCESSOR BY THE NUMBER OF LINES IT REQUIRES. BABY PROCESSORS USE 4 BITS, MOST SYSTEMS USE 8 BITS & LATEST PROCESSORS ARE 16 OR 32 BIT!!

BY TURNING ON COMBINATIONS OF LINES WE CAN GET 256 DIFFERENT VALUES. THE LOWEST VALUE IS 00000000. THE HIGHEST IS 11111111.

MICROPROCESSOR SYSTEMS OPERATE ON BINARY VALUES SUCH AS: 10111001, 00110111, 10001100 OR 01010110 etc.

EACH BINARY DIGIT IS CALLED A BIT:



A GROUP OF 8 BITS IS CALLED A BYTE.
HALF A BYTE IS CALLED A NIBBLE.

COMPUTERS ACCEPT INFORMATION (DATA) ONE BYTE AT A TIME

WORDS THE SIZE OF A WORD DEPENDS ON THE SIZE OF A COMPUTER.

A 4 BIT MICRO	OPERATES	WITH	4-BIT	WORDS
8 "	"	"	"	8-BIT	"
16 "	"	"	"	16-BIT	"

WRITING A PROGRAM

WHEN WRITING A MACHINE CODE PROGRAM THERE ARE 3 POINTS YOU MUST REMEMBER:

1. EACH STEP MUST BE VERY SMALL. THE MICRO IS CAPABLE OF PERFORMING ONLY A VERY SIMPLE TASK AT ANY ONE TIME.
2. EACH STEP MUST BE ABLE TO BE CONVERTED TO A MACHINE-CODE INSTRUCTION.
3. THE PROCESSOR WILL BE RUNNING ALL THE TIME & THUS PROGRAMS MUST BE EITHER A LOOP OR INCLUDE DELAYS TO SLOW DOWN THEIR EXECUTION RATE TO SUIT HUMAN INVOLVEMENT.

THIS IS THE SKILL IN WRITING A PROGRAM. TO CONVERT AN IDEA INTO A SERIES OF SMALL STEPS WHICH CAN BE EXECUTED BY A MICRO.

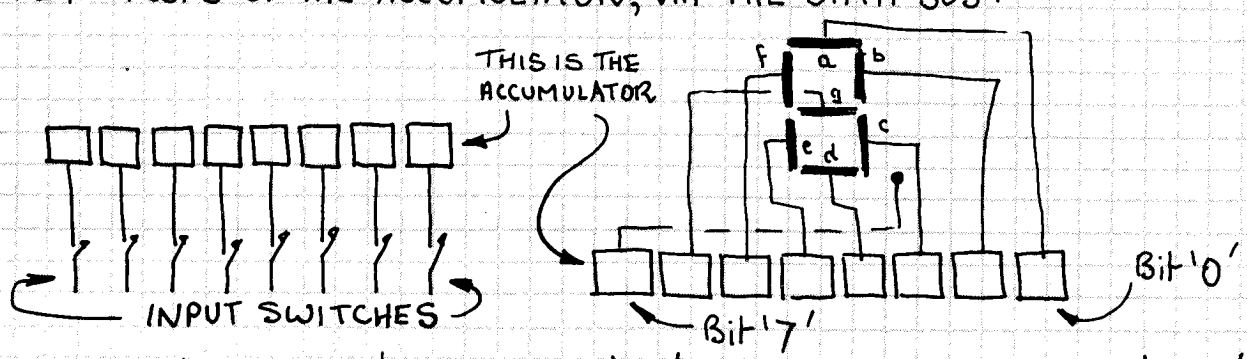
HERE IS A SIMPLE REQUIREMENT: TO TURN ON SEGMENTS OF A 7-SEGMENT DISPLAY VIA A SET OF INPUT SWITCHES.

THE PROGRAM WILL NEED TO BE A LOOP & THE MICRO WILL EXECUTE IT MANY TIMES PER SECOND.

```
IN A(01)      DB 01
OUT(02)A      D3 02
JR            18 FA
```

THE SWITCHES ARE SCANNED & THEIR VALUE LOADED INTO THE ACCUMULATOR.
THE VALUE (IN THE ACCUMULATOR) IS OUTPUTTED TO THE 7-SEGMENT DISPLAY.
THE PROGRAM JUMPS TO THE BEGINNING & REPEATS THE SEQUENCE.

EACH INPUT SWITCH IS INDIRECTLY CONNECTED TO ONE OF THE FLIP-FLOPS OF THE ACCUMULATOR, VIA THE DATA BUS:



THEY ARE 'LOOKED AT' DURING THE 'IN' INSRUCTION. WHEN THE 'OUT' INSTRUCTION IS EXECUTED THE VALUE IN EACH FLIPFLOP WILL BE PASSED TO THE SEGMENTS OF THE DISPLAY.

IF Bit 0 IS SET, SEGMENT 'a' WILL ILLUMINATE. THE SAME APPLIES TO BITS 1,2,3,4,5,6 & 7. (BIT 7 ILLUMINATES THE DECIMAL POINT).

TO BLINK THE DISPLAY

WE CAN COMBINE OUR KNOWLEDGE OF **OUT & DJNZ** IN A PROGRAM WHICH BLINKS THE DISPLAY.

AIM: TO TURN THE DISPLAY ON & OFF.

THIS IS A LOOP PROGRAM WHICH OUTPUTS A VALUE TO A DISPLAY FOR A DELAY PERIOD AND THEN A ZERO VALUE FOR A DELAY PERIOD. THE PROGRAM THEN REPEATS.

```
LD B,00        06 00
LD A,06        3E 06
OUT (02)A      D3 02
DJNZ FE        10 FE
DJNZ FE        10 FE
DJNZ FE        10 FE
LD A,00        3E 00
OUT (02)A      D3 02
DJNZ FE        10 FE
DJNZ FE        10 FE
DJNZ FE        10 FE
JR EA          18 EA
```

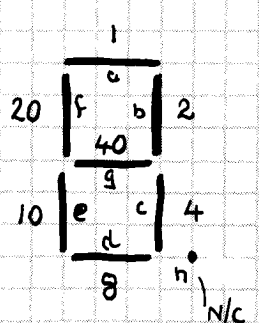
THIS IS WHERE THE COMPUTER IS SPENDING TIME DOING "NOTHING" (EXCEPT DECREMENTING REGISTER B) WHILE THE DISPLAY REMAINS ILLUMINATED (OR BLANK) SO THAT THE HUMAN EYE CAN DETECT THE 'BLINKING' EFFECT.

THE BLINK RATE WILL DEPEND ON THE FREQUENCY OF THE CLOCK AND YOU CAN ADD MORE DJNZ'S IF REQUIRED.

ONLY ONE LD B,00 IS REQUIRED AND IT IS AT THE START OF THE PROGRAM. AT THE COMPLETION OF A DJNZ OPERATION REGISTER B IS LEFT IN A ZERO STATE AND IS READY FOR THE NEXT DJNZ.

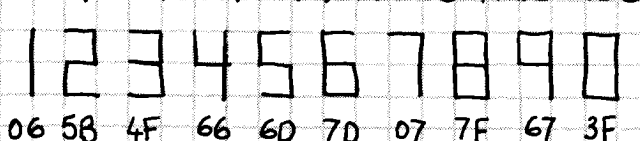
THE ACCUMULATOR IS LOADED WITH 06 IN THE PROGRAM BUT CAN BE LOADED WITH ANY VALUE FROM 01 TO FF.

THE VALUE YOU CHOOSE WILL DEPEND UPON HOW THE DISPLAY IS WIRED & WHAT YOU WANT TO APPEAR. THE MICROCOMP (SEE INSIDE COVER OF THIS ISSUE) HAS THE FOLLOWING SEGMENT VALUES:



YOU WILL NOTICE THE VALUES INCREASE ACCORDING TO THE LETTERING OF THE SEGMENTS & ARE EASY TO REMEMBER.

TO PRODUCE LETTERS OR NUMBERS ON THE DISPLAY THE HEX. VALUES ARE ADDED EG.



TO INCREMENT THE DISPLAY

THERE ARE TWO WAYS OF INCREMENTING THE DISPLAY. ONE IS BINARY INCREMENT, THE OTHER IS NUMERICAL INCREMENT.

BINARY INCREMENT IS THE SIMPLEST & WILL PRODUCE 128 DIFFERENT PATTERNS ON THE SCREEN. SOME OF THESE WILL NOT MAKE ANY SENSE BUT OTHERS WILL PRODUCE A LETTER OR NUMBER.

BEFORE WE CAN CONSIDER A BINARY INCREMENT PROGRAM WE MUST CONSIDER A PROBLEM CALLED DEBOUNCE. WE KNOW THAT SWITCHES MUST BE DEBOUNCED WHEN CONNECTED TO AN ELECTRONIC CIRCUIT — THE SAME APPLIES TO INPUT DEVICES CONNECTED TO A COMPUTER.

THE REASON IS THIS: PROGRAMS OPERATE AT SUCH A HIGH SPEED THAT THE PRESS OF A BUTTON MAY BE DETECTED MORE THAN ONCE DUE TO THE PROGRAM LOOPING VERY QUICKLY. THIS WILL GIVE AN INACCURATE COUNT.

TO PREVENT THIS FROM OCCURING WE MUST PRODUCE A PROGRAM CONTAINING 2 SMALL LOOPS. ONE DETECTS WHEN THE BUTTON IS PRESSED & THE OTHER DETECTS WHEN THE BUTTON IS NOT PRESSED.

THE MICRO LOOPS AROUND ONE OF THESE ACCORDING TO THE STATE OF THE BUTTON AND WHEN A COMPLETE CYCLE IS COMPLETED, THE DISPLAY INCREMENTS BY A COUNT-OF-ONE.

HERE IS THE PROGRAM TO ACHIEVE THIS:*

```
XOR A          AF
LD C,A         4F
IN A,(01)      DB 01  } THE PROGRAM LOOPS HERE IF THE
BIT 7,A        CB 7F  } BUTTON IS NOT PRESSED. WHEN
JR Z           28 FA  } BUTTON IS PRESSED PROGRAM ADVANCES
INC C          0C
LD A,C         79
OUT (02) A     D3 02
IN A (01)      DB 01  } PROGRAM LOOPS HERE WHILE BUTTON
BIT 7,A        CB 7F  } IS PRESSED AND ADVANCES WHEN
JR NZ          20 FA  } BUTTON RELEASED.
JR             18 EE
```

THE ACCUMULATOR IS USED FOR 2 FUNCTIONS. IT OUTPUTS THE VALUE OF THE COUNT & LOOKS TO SEE IF THE SWITCH IS PRESSED. THAT'S WHY WE NEED ANOTHER REGISTER TO HOLD THE VALUE OF THE COUNT.

THE SWITCH IS CONNECTED TO LINE 8 OF THE INPUT PORT (BIT 7), AND THE PROGRAM LOOKS AT BIT 7 TO SEE IF IT IS 'SET' OR 'RESET'.

*THIS PROGRAM HAS BEEN TAKEN FROM THE PROGRAMMED EPROM IN THE MICROCOMP COMPUTER — DESIGNED BY TALKING ELECTRONICS FOR THE TEACHING OF MACHINE CODE PROGRAMMING.

0-9 COUNTER PROGRAM

LD C,0A	0E 0A
LD DE 0080	11 00 08
IN A,(01)	DB 01
BIT 7, A	CB 7F
JR Z	28 FA
INC DE	13
LD A(DE)	1A
OUT (02) A	D3 02
IN A (01)	DB 01
BIT 7, A	CB 7F
JRNZ	20 FA
DEC C	0D
JR Z	28 E8
JR	18 E5

BYTE TABLE AT 0080:

3F	= 0
06	= 1
5B	= 2
4F	= 3
66	= 4
6D	= 5
7D	= 6
07	= 7
7F	= 8
67	= 9

LD C,0A — C IS THE 'BYTE-COUNT' REGISTER. THE BYTE TABLE CONTAINS 10 BYTES. THIS IS 0A IN HEXADECIMAL.

LD DE,0080 — THE POINTER REGISTER DE IS LOADED WITH THE START ADDRESS OF THE BYTE TABLE. THE LOW BYTE IS PLACED FIRST IN THE PROGRAM AND THE HIGH BYTE SECOND.

IN A (01)
BIT 7, A
JR Z — THE SWITCH IS LOOKED AT AND IF IT IS NOT PUSHED, THE PROGRAM LOOPS AROUND THE 3 INSTRUCTIONS. IF THE BUTTON IS PUSHED, BIT 7 OF THE ACCUMULATOR WILL BE '1' & THE PROGRAM WILL ADVANCE TO THE NEXT INSTRUCTION.

INC DE — THE POINTER REGISTER PAIR WILL ADVANCE TO THE NEXT ADDRESS (EG: "06")

LD A(DE) — THE ACCUMULATOR IS LOADED WITH THE VALUE FOUND AT THE ADDRESS POINTED TO BY DE.

OUT (02)A — THE VALUE (EG "06") IS OUTPUTTED TO THE DISPLAY.

IN A (01)
BIT 7, A
JRNZ — THESE 3 INSTRUCTIONS FORM A LOOP WHICH IS EXECUTED SO LONG AS THE BUTTON IS PRESSED

DEC C — THE BYTE-COUNT REGISTER IS DECREMENTED AND THE ZERO FLAG IS SET WHEN THE 'C' REGISTER BECOMES ZERO.

JR Z. — THE PROGRAMS JUMPS IF THE ZERO FLAG IS 'SET'.

JR — THIS IS AN UNCONDITIONAL JUMP WHICH IS EXECUTED FOR EACH LOOP OF THE PROGRAM EXCEPT WHEN THE END OF THE TABLE IS DETECTED.

END

0-9 COUNTER

PRODUCING CHARACTERS SUCH AS 0, 1, 2, 3 ETC ON A DISPLAY REQUIRES A TABLE. THE MICRO STEPS THROUGH THIS TABLE AND WE RECOGNISE THIS AS COUNTING.

AS FAR AS THE MICRO IS CONCERNED IT IS "LOOKING THRU A TABLE" — IT DOESN'T RECOGNISE IT AS COUNTING.

THE VALUES COULD BE WRITTEN IN THE REVERSE DIRECTION OR MIXED-UP & THE MICRO WILL STILL BE "LOOKING THRU A TABLE!"

WHEN WE PRODUCE A TABLE WE MUST TELL THE MICRO TWO THINGS: THESE ARE: THE START ADDRESS & HOW MANY BYTES IT CONTAINS.

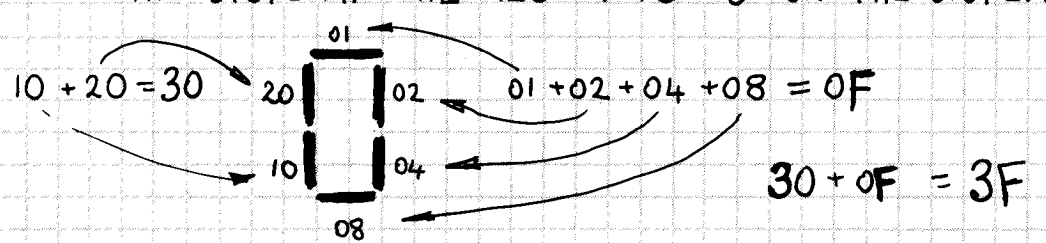
THE STRUCTURE OF THIS 0-9 PROGRAM IS IDENTICAL TO THAT ON THE PREVIOUS PAGE EXCEPT THE INCREMENT FUNCTION IS HANDLED BY A POINTER REGISTER WHICH POINTS TO AN ADDRESS IN THE TABLE. THE VALUE AT THIS ADDRESS IS LOADED INTO THE ACCUMULATOR & DISPLAYED.

THIS IS HOW IT IS DONE: ① REGISTER DE IS LOADED WITH 0080

0080 3F

② AT 0080, THE VALUE 3F IS FOUND.

③ 3F IS LOADED INTO THE ACCUMULATOR AND OUTPUT TO THE DISPLAY. THE RESULT IS '0' ON THE DISPLAY.



THE PROGRAM MUST INCLUDE AN INSTRUCTION WHICH DETECTS THE END OF THE TABLE. THIS IS DONE BY LOADING A REGISTER WITH A VALUE & PROGRESSIVELY DECREMENTING IT UNTIL IT REACHES ZERO.

ZERO IS A VERY HANDY VALUE TO DETECT AS THERE ARE MACHINE CODE INSTRUCTIONS FOR THIS. WHEN THE REGISTER REACHES ZERO THE PROGRAM IS INSTRUCTED TO RE-START & THE REGISTER IS RE-LOADED.

THE POINTER REGISTER IS DE (IN OUR CASE) AND IT IS LOADED WITH THE STARTING ADDRESS OF THE TABLE. ON EACH PASS OF THE PROGRAM THIS REGISTER-PAIR LOOKS AT THE NEXT LOWER ADDRESS — UNTIL THE END OF THE TABLE.

CONT.....

DJNZ 10 dis

THIS IS A SPECIAL TWO BYTE INSTRUCTION USING THE B REGISTER. IT IS MAINLY USED TO PRODUCE A SHORT DELAY & OPERATES IN THE FOLLOWING MANNER:

- THE B REGISTER MUST BE PRE-LOADED WITH A START VALUE (00 – FF)
- A JUMP RELATIVE VALUE MUST BE ASSIGNED TO THE SECOND BYTE.
- THE INSTRUCTION IS THEN ADDED TO THE PROGRAM.

THE MACHINE CODE VALUE FOR DJNZ IS "10".

THE DISPLACEMENT VALUE MUST BE A "JUMP BACK" SO THAT THE INSTRUCTION PRODUCES A LOOP. IT CAN RANGE FROM FE TO 80. ANY VALUES LOWER THAN THIS WILL PRODUCE A FORWARD JUMP & THE DJNZ INSTRUCTION WILL NOT BE EXECUTED – APART FROM THE FACT THAT BYTES AFTER THE INSTRUCTION WILL BE JUMPED OVER!

A TYPICAL VALUE FOR THE DISPLACEMENT BYTE IS FE. THIS CAUSES THE PROCESSOR TO JUMP TO THE BEGINNING OF THE DJNZ INSTRUCTION & EXECUTE LOOPS UNTIL THE B REGISTER IS ZERO.

```
EG:   LD B,FF      06 FF
      LD A,08      3E 08
      OUT (02),A   D3 02
      DJNZ FE      10 FE
```

THE PROGRAM WILL: LOAD B WITH A DECIMAL VALUE OF 255 (FF). LOAD THE ACCUMULATOR WITH 08. OUTPUT 08 TO PORT 2. CREATE A DELAY OF 255 LOOPS OF DECREMENTING THE B REGISTER.

THE DELAY IS CREATED BY THE NUMBER OF CLOCK CYCLES REQUIRED TO CARRY OUT EACH DJNZ INSTRUCTION. ONE LOOP OF DJNZ WILL TAKE 13 CLOCK CYCLES & WHEN MULTIPLIED BY 255, A DELAY OF 3315 CYCLES IS CREATED.

THE B REGISTER IS LEFT IN A ZERO STATE AT THE END OF THE LOOPING AND THIS IS IDEAL FOR THE NEXT TIME A DJNZ INSTRUCTION IS REACHED.

LOADING B WITH ZERO CREATES THE LONGEST DELAY POSSIBLE & CAN BE EXPLAINED AS FOLLOWS:

EACH TIME DJNZ IS EXECUTED, THE FIRST PART OF THE OPERATION IS TO DECREMENT THE B REGISTER. IF B IS 00, THE DECREMENT OPERATION CAUSES B TO GO TO FF! THE Z-80 DETECTS B IS NOT ZERO AND JUMPS BACK TO THE ADDRESS AS INSTRUCTED BY THE DISPLACEMENT BYTE. AND IT CONTINUES TO LOOP 256 TIMES!

DJNZ'S CAN BE PLACED AFTER ONE ANOTHER TO CREATE LONGER DELAYS, THUS:

```
DJNZ  10 FE
DJNZ  10 FE
DJNZ  10 FE
```

IF B IS LOADED WITH 80, A SHORTER DELAY WILL BE CREATED THUS:

```
LD B 80        06 80
DJNZ FE        10 FE
```

IF MORE THAN ONE DJNZ IS USED ONLY THE FIRST DJNZ WILL BE SHORT AS REGISTER B WILL BE ZERO AT THE BEGINNING OF THE 2ND & 3RD DJNZ.

ANOTHER WAY OF PRODUCING A LONGER DELAY IS TO JUMP BACK OVER A NUMBER OF PREVIOUS INSTRUCTIONS THUS:

```
LD B 00        06 00
LD A 08        3E 08
OUT (02) A     D3 02
DJNZ FA        10 FA
```

IF YOU JUMP BACK TO THUS:

```
LD B,00        06 00
LD A 08        3E 08
OUT (02) A     D3 02
DJNZ F8        10 F8
```

REGISTER B WILL BE CONSTANTLY RE-LOADED AT THE END OF EACH LOOP AND WILL NEVER DECREMENT TO ZERO & THE MICRO WILL NOT GET OUT OF THE LOOP!

WE HAVE SHOWN THE LONGEST DELAY IS 00. THE SHORTEST DELAY IS 01 – WHICH WILL PRODUCE ONLY ONE LOOP.

NESTED LOOP

A NESTED LOOP CAN BE CREATED WITH THE DJNZ INSTRUCTION & ANOTHER REGISTER, AS FOLLOWS:

```
LD B 00
LD C 20
DJNZ FE
DEC C
JRNZ
```

THE FIRST TWO INSTRUCTIONS ARE CALLED "SET UP" & IT IS IMPORTANT NOT TO JUMP BACK TO THESE OTHERWISE THE LOOP(S) WILL NEVER BE DECREMENTED TO ZERO.

THE PROGRAM WILL PERFORM 256 LOOPS OF DECREMENTING B THEN ADVANCE TO "DEC C". IF C IS NOT ZERO THE PROGRAM WILL JUMP BACK TO DJNZ. THUS IT WILL PERFORM 20H LOOPS OF DJNZ. [20H LOOPS = 32 LOOPS]. THIS PROGRAM IS A LOOP WITHIN A LOOP OR NESTED LOOP & IS VERY HANDY FOR PRODUCING LONG TIME DELAYS.

Z80 Machine Codes

This table contains over 700 Machine Code instructions for the Z80.
It has been compiled from Zilog Data sheets, SGS Data books, Z80 Programming by P. Levison (now out of print) and Micro-Professor Programming Handbooks.
Two books to help with the interpretation of this table are: **Z80 ASSEMBLY LANGUAGE** by Lance A. Leventhal. (Mc Graw Hill), and **PR8GRAMMING the Z80** by Rodnay Zaks. (Sybex).

ADC A,(HL)	8E
ADC A,(IX+dis)	DD BE XX
ADC A,(IY+dis)	FD 8E XX
ADC A,A	8F
ADC A,B	88
ADC A,C	89
ADC A,D	8A
ADC A,dd	CE dd
ADC A,E	8B
ADC A,H	8C
ADC A,L	8D
ADC HL,BC	ED 4A
ADC HL,DE	FD 5A
ADC HL,HL	FD 6A
ADC HL,SP	ED 7A
ADD A,(HL)	86
ADD A,(IX+dis)	DD 86 XX
ADD A,(IY+dis)	FD 86 XX
ADD A,A	B7
ADD A,B	80
ADD A,C	81
ADD A,D	82
ADD A,dd	C6 dd
ADD A,E	83
ADD A,H	84
ADD A,L	85
ADD HL,BC	09
ADD HL,DE	19
ADD HL,HL	29
ADD HL,SP	39
ADD IX,BC	DD 09
ADD IX,DE	DD 19
ADD IX,IX	DD 29
ADD IX,SP	DD 39
ADD IY,BC	FD 09
ADD IY,DE	FD 19
ADD IY,IY	FD 29
ADD IY,SP	FD 39
AND (HL)	A6
AND (IX+dis)	DD A6 XX
AND (IY+dis)	FD A6 XX
AND A	A7
AND B	A0
AND C	A1
AND D	A2
AND dd	E6 dd
AND E	A3
AND H	A4
AND L	A5
BIT 0,(HL)	CB 46
BIT 0,(IX+dis)	DD CB XX 46
BIT 0,(IY+dis)	FD CB XX 46
BIT 0,A	CB 47
BIT 0,B	CB 40
BIT 0,C	CB 41
BIT 0,D	CB 42
BIT 0,E	CB 43
BIT 0,H	CB 44
BIT 0,L	CB 45
BIT 1,(HL)	CB 4E
BIT 1,(IX+dis)	DD CB XX 4E
BIT 1,(IY+dis)	FD CB XX 4E
BIT 1,A	CB 4F
BIT 1,B	CB 48
BIT 1,C	CB 49
BIT 1,D	CB 4A
BIT 1,E	CB 4B
BIT 1,H	CB 4C
BIT 1,L	CB 4D
BIT 2,(HL)	CB 56
BIT 2,(IX+dis)	DD CB XX 56
BIT 2,(IY+dis)	FD CB XX 56
BIT 2,A	CB 57
BIT 2,B	CB 50
BIT 2,C	CB 51
BIT 2,D	CB 52
BIT 2,E	CB 53
BIT 2,H	CB 54
BIT 2,L	CB 55
BIT 3,(HL)	CB 5E
BIT 3,(IX+dis)	DD CB XX 5E
BIT 3,(IY+dis)	FD CB XX 5E
BIT 3,A	CB 5F
BIT 3,B	CB 58
BIT 3,C	CB 59
BIT 3,D	CB 5A
BIT 3,E	CB 5B
BIT 3,H	CB 5C
BIT 3,L	CB 5D
BIT 4,(HL)	CB 66
BIT 4,(IX+dis)	DD CB XX 66
BIT 4,(IY+dis)	FD CB XX 66
BIT 4,A	CB 67
BIT 4,B	CB 60
BIT 4,C	CB 61
BIT 4,D	CB 62
BIT 4,E	CB 63
BIT 4,H	CB 64
BIT 4,L	CB 65
BIT 5,(HL)	CB 6E
BIT 5,(IX+dis)	DD CB XX 6E
BIT 5,(IY+dis)	FD CB XX 6E
BIT 5,A	CB 6F
BIT 5,B	CB 68
BIT 5,C	CB 69
BIT 5,D	CB 6A
BIT 5,E	CB 6B
BIT 5,H	CB 6C
BIT 5,L	CB 6D
BIT 6,(HL)	CB 76
BIT 6,(IX+dis)	DD CB XX 76
BIT 6,(IX+dis)	FD CB XX 76
BIT 6,A	CB 77
BIT 6,B	CB 70
BIT 6,C	CB 71
BIT 6,D	CB 72
BIT 6,E	CB 73
BIT 6,H	CB 74
BIT 6,L	CB 75
BIT 7,(HL)	CB 7E
BIT 7,(IX+dis)	DD CB XX 7E
BIT 7,(IY+dis)	FD CB XX 7E
BIT 7,A	CB 7F
BIT 7,B	CB 78
BIT 7,C	CB 79
BIT 7,D	CB 7A
BIT 7,E	CB 7B
BIT 7,H	CB 7C
BIT 7,L	CB 7D
CALL ADDR	CD XX XX
CALL C,ADDR	DC XX XX
CALL M,ADDR	FC XX XX
CALL NC,ADDR	D4 XX XX
CALL NZ,ADDR	C4 XX XX
CALL P,ADDR	F4 XX XX
CALL PE,ADDR	EC XX XX
CALL PO,ADDR	E4 XX XX
CALL Z,ADDR	CC XX XX
CCF	3F
CP (HL)	BE
CP (IX+dis)	DD BE XX
CP (IY+dis)	FD BE XX
CP A	BF
CP B	B8
CP C	B9
CP D	BA
CP dd	FE dd
CP E	BB
CP H	BC
CP L	BD
CPD	ED A9
CPDR	ED B9
CPI	ED A1
CPIR	ED B1
CPL	2F
DAA	27
DEC (HL)	35
DEC (IX+dis)	DD 35 XX
DEC (IY+dis)	FD 35 XX
DEC A	3D
DEC B	05
DEC BC	0B
DEC C	0D
DEC D	15
DEC DE	1B
DEC E	1D
DEC H	25
DEC HL	2B
DEC IX	DD 2B
DEC IY	FD 2B
DEC L	2D
DEC SP	3B
DI	F3
DJNZ dis	10 XX
EI	FB
EX (SP),HL	E3
EX (SP),IX	DD E3
EX (SP),IY	FD E3
EX AF,AF'	08
EX DE,HL	EB
EXX	D9
HALT	76
IM 0	ED 46
IM 1	ED 56
IM 2	ED 5E
IN A,(C)	ED 78
IN A,port	DB XX
IN B,(C)	ED 40
IN C,(C)	ED 48
IN D,(C)	ED 50
IN E,(C)	ED 58
IN H,(C)	ED 60
IN L,(C)	ED 6B
INC (HL)	34
INC (IX+dis)	DD 34 XX
INC (IY+dis)	FD 34 XX
INC A	3C
INC B	04
INC BC	03
INC C	0C
INC D	14
INC DE	13
INC E	1C
INC H	24
INC HL	23
INC IX	DD 23
INC IY	FD 23
INC L	2C
INC SP	33
IND	ED AA
INDR	ED BA
INI	ED A2
INIR	ED B2
JP (HL)	E9
JP (IX)	DD E9
JP (IY)	FD E9
JP ADDR	C3 XX XX
JP C,ADDR	DA XX XX
JP M,ADDR	FA XX XX
JP NC,ADDR	D2 XX XX
JP NZ,ADDR	C2 XX XX
JP P,ADDR	F2 XX XX
JP PE,ADDR	EA XX XX
JP PO,ADDR	E2 XX XX
JP Z,ADDR	CA XX XX
JR C,dis	38 XX
JR dis	18 XX
JR NC,dis	30 XX
JR NZ,dis	20 XX
JR Z,dis	28 XX
LD (ADDR),A	32 XX XX
LD (ADDR),BC	ED 43 XX XX
LD (ADDR),DE	ED 53 XX XX
LD (ADDR),HL	ED 63 XX XX
LD (ADDR),HL	22 XX XX
LD (ADDR),IX	DD 22 XX XX
LD (ADDR),IY	FD 22 XX XX
LD (ADDR),SP	ED 73 XX XX
LD (BC),A	02
LD (DE),A	12
LD (HL),A	77
LD (HL),B	70
LD (HL),C	71
LD (HL),D	72
LD (HL),dd	36 dd
LD (HL),E	73
LD (HL),H	74
LD (HL),L	75
LD (IX+dis),A	DD 77 XX
LD (IX+dis),B	DD 70 XX
LD (IX+dis),C	DD 71 XX
LD (IX+dis),D	DD 72 XX
LD (IX+dis),dd	DD 36 XX dd
LD (IX+dis),E	DD 73 XX
LD (IX+dis),H	DD 74 XX
LD (IX+dis),L	DD 75 XX
LD (IY+dis),A	FD 77 XX
LD (IY+dis),B	FD 70 XX
LD (IY+dis),C	FD 71 XX
LD (IY+dis),D	FD 72 XX
LD (IY+dis),dd	FD 36 XX dd
LD (IY+dis),E	FD 73 XX
LD (IY+dis),H	FD 74 XX
LD (IY+dis),L	FD 75 XX
LD A,(ADDR)	3A XX XX
LD A,(BC)	0A
LD A,(DE)	1A
LD A,(HL)	7E
LD A,(IX+dis)	DD 7E XX
LD A,(IY+dis)	FD 7E XX
LD A,A	7F
LD A,B	7B
LD A,C	79
LD A,D	7A
LD A,dd	3E dd
LD A,E	7B
LD A,H	7C
LD A,I	ED 57
LD A,L	7D
LD A,R	ED 5F
LD B,(HL)	46
LD B,(IX+dis)	DD 46 XX
LD B,(IY+dis)	FD 46 XX
LD B,A	47
LD B,B	40
LD B,C	41
LD B,D	42
LD B,dd	06 dd
LD B,E	43
LD B,H	44
LD B,L	45
LD BC,(ADDR)	ED 4B XX XX
LD BC,dd	01 dd dd
LD C,(HL)	4E
LD C,(IX+dis)	DD 4E XX
LD C,(IY+dis)	FD 4E XX
LD C,A	4F
LD C,B	48
LD C,C	49
LD C,D	4A
LD C,dd	DE dd
LD C,E	4B
LD C,H	4C
LD C,L	4D
LD D,(HL)	56
LD D,(IX+dis)	DD 56 XX
LD D,(IY+dis)	FD 56 XX
LD D,A	57
LD D,B	50
LD D,C	51
LD D,D	52
LD D,dd	16 dd
LD D,E	53
LD D,H	54
LD D,L	55
LD DE,(ADDR)	ED 5B XX XX
LD DE,dddd	11 dd dd
LD E,(HL)	5E
LD E,(IX+dis)	DD 5E XX
LD E,(IY+dis)	FD 5E XX
LD E,A	5F
LD E,B	5B
LD E,C	59
LD E,D	5A
LD E,dd	1E dd
LD E,E	5B
LD E,H	5C
LD E,L	5D
LD H,(HL)	66
LD H,(IX+dis)	DD 66 XX
LD H,(IY+dis)	FD 66 XX
LD H,A	67
LD H,B	60
LD H,C	61
LD H,D	62
LD H,dd	26 dd
LD H,E	63
LD H,H	64
LD H,L	65
LD HL,(ADDR)	ED 6B XX XX
LD HL,(ADDR)	2A XX XX
LD HL,dddd	21 dd dd
LD I,A	ED 47
LD IX,(ADDR)	DD 2A XX XX
LD IX,dddd	DD 21 dd dd
LD IY,(ADDR)	FD 2A XX XX
LD IY,dddd	FD 21 dd dd
LD L,(HL)	6E
LD L,(IX+dis)	DD 6E XX
LD L,(IY+dis)	FD 6E XX
LD L,A	6F
LD L,B	68
LD L,C	69
LD L,D	6A
LD L,dd	2E dd
LD L,E	6B
LD L,H	6C
LD L,L	6D
LD R,A	ED 4F
LD SP,(ADDR)	ED 7B XX XX
LD SP,dddd	31 dd dd
LD SP,HL	F9
LD SP,IX	DD F9
LD SP,IY	FD F9
LDD	ED A8
LDDR	ED B8
LDI	ED A0
LDIR	ED B0
NEG	ED 44
NOP	00
OR (HL)	B6
OR (IX+dis)	DD B6 XX
OR (IY+dis)	FD B6 XX
OR A	B7
OR B	B0
OR C	B1
OR D	B2
OR dd	F6 dd
OR E	B3
OR H	B4
OR L	B5
OTDR	ED BB
OTIR	ED B3
OUT (C),A	ED 79
OUT (C),B	ED 41
OUT (C),C	ED 49
OUT (C),D	ED 51
OUT (C),E	ED 59
OUT (C),H	ED 61
OUT (C),L	ED 69
OUT port,A	D3 port
OUTD	ED AB
OUTI	ED A3
POP AF	F1
POP BC	C1
POPDE	D1
POPHL	E1
POPIX	DD E1
POPIY	FD E1
PUSH AF	F5
PUSH BC	C5
PUSH DE	D5
PUSH HL	E5
PUSH IX	DD E5
PUSH IY	FD E5
RES 0,(HL)	CB 86
RES 0,(IX+dis)	DD CB XX B6
RES 0,(IY+dis)	FD CB XX B6
RES 0,A	CB B7
RES 0,B	CB B0
RES 0,C	CB B1
RES 0,D	CB B2
RES 0,E	CB B3
RES 0,H	CB B4
RES 0,L	CB B5
RES 1,(HL)	CB 8E
RES 1,(IX+dis)	DD CB XX 8E
RES 1,(IY+dis)	FD CB XX 8E
RES 1,A	CB 8F
RES 1,B	CB B8
RES 1,C	CB B9
RES 1,D	CB 8A
RES 1,E	CB 8B
RES 1,H	CB 8C
RES 1,L	CB 8D
RES 2,(HL)	CB 96
RES 2,(IX+dis)	DD CB XX 96
RES 2,(IY+dis)	FD CB XX 96
RES 2,A	CB 97
RES 2,B	CB 90
RES 2,C	CB 91
RES 2,D	CB 92
RES 2,E	CB 93
RES 2,H	CB 94
RES 2,L	CB 95
RES 3,(HL)	CB 9E
RES 3,(IX+dis)	DD CB XX 9E
RES 3,(IY+dis)	FD CB XX 9E
RES 3,A	CB 9F
RES 3,B	CB 98
RES 3,C	CB 99
RES 3,D	CB 9A
RES 3,E	CB 9B
RES 3,H	CB 9C
RES 3,L	CB 9D
RES 4,(HL)	CB A6
RES 4,(IX+dis)	DD CB XX A6
RES 4,(IY+dis)	FD CB XX A6
RES 4,A	CB A7
RES 4,B	CB A0
RES 4,C	CB A1
RES 4,D	CB A2
RES 4,E	CB A3
RES 4,H	CB A4
RES 4,L	CB A5
RES 5,(HL)	CB AE
RES 5,(IX+dis)	DD CB XX AE
RES 5,(IY+dis)	FD CB XX AE
RES 5,A	CB AF
RES 5,B	CB AB
RES 5,C	CB A9
RES 5,D	CB AA
RES 5,E	CB AB
RES 5,H	CB AC
RES 5,L	CB AD
RES 6,(HL)	CB B6
RES 6,(IX+dis)	DD CB XX B6
RES 6,(IY+dis)	FD CB XX B6
RES 6,A	CB B7
RES 6,B	CB B0
RES 6,C	CB B1
RES 6,D	CB B2
RES 6,E	CB B3
RES 6,H	CB B4
RES 6,L	CB B5
RES 7,(HL)	CB BE
RES 7,(IX+dis)	DD CB XX BE
RES 7,(IY+dis)	FD CB XX BE
RES 7,A	CB BF
RES 7,B	CB B8
RES 7,C	CB B9
RES 7,D	CB BA
RES 7,E	CB BB
RES 7,H	CB BC
RES 7,L	CB BD
RET	C9
RET C	D8
RET M	F8
RET NC	D0
RET NZ	C0
RET P	F0
RET PE	E8
RET PO	E0
RET Z	C8
RETI	ED 4D
RETN	ED 45
RL (HL)	CB 16
RL (IX+dis)	DD CB XX 16
RL (IY+dis)	FD CB XX 16
RL A	CB 17
RL B	CB 10
RL C	CB 11
RL D	CB 12
RL E	CB 13
RL H	CB 14
RL L	CB 15
RLA	17
RLC (HL)	CB 06
RLC (IX+dis)	DD CB XX 06
RLC (IY+dis)	FD CB XX 06
RLC A	CB 07
RLC B	CB 00
RLC C	CB 01
RLC D	CB 02
RLC E	CB 03
RLC H	CB 04
RLC L	CB 05
RLCA	07
RLD	ED 6F
RR (HL)	CB 1E
RR (IX+dis)	DD CB XX 1E
RR (IY+dis)	FD CB XX 1E
RR A	CB 1F
RR B	CB 18
RR C	CB 19
RR D	CB 1A
RR E	CB 1B
RR H	CB 1C
RR L	CB 1D
RRA	1F
RRC (HL)	CB 0E
RRC (IX+dis)	DD CB XX 0E
RRC (IY+dis)	FD CB XX 0E
RRC A	CB 0F
RRC B	CB 0B
RRC C	CB 09
RRC D	CB 0A
RRC E	CB 0B
RRC H	CB 0C
RRC L	CB 0D
RRCA	0F
RRD	ED 67
RST 00	C7
RST 08	CF
RST 10	D7
RST 18	DF
RST 20	E7
RST 28	EF
RST 30	F7
RST 38	FF
SBC A,(HL)	9E
SBC A,(IX+dis)	DD 9E XX
SBC A,(IY+dis)	FD 9E XX
SBC A,A	9F
SBC A,B	9B
SBC A,C	99
SBC A,D	9A
SBC A,dd	DE dd
SBC A,E	9B
SBC A,H	9C
SBC A,L	9D
SBC HL,BC	ED 42
SBC HL,DE	ED52
SBC HL,HL	ED 62
SBC HL,SP	ED 72
SCF	37
SET 0,(HL)	CB C6
SET 0,(IX+dis)	DD CB XX C6
SET 0,(IY+dis)	FD CB XX C6
SET 0,A	CB C7
SET 0,B	CB C0
SET 0,C	CB C1
SET 0,D	CB C2
SET 0,E	CB C3
SET 0,H	CB C4
SET 0,L	CB C5
SET 1,(HL)	CB CE
SET 1,(IX+dis)	DD CB XX CE
SET 1,(IY+dis)	FD CB XX CE
SET 1,A	CB CF
SET 1,B	CB C8
SET 1,C	CB C9
SET 1,D	CB CA
SET 1,E	CB CB
SET 1,H	CB CC
SET 1,L	CB CD
SET 2,(HL)	CB D6
SET 2,(IX+dis)	DD CB XX D6
SET 2,(IY+dis)	FD CB XX D6
SET 2,A	CB D7
SET 2,B	CB D0
SET 2,C	CB D1
SET 2,D	CB D2
SET 2,E	CB D3
SET 2,H	CB D4
SET 2,L	CB D5
SET 3,(HL)	CB DE
SET 3,(IX+dis)	DD CB XX DE
SET 3,(IY+dis)	FD CB XX DE
SET 3,A	CB DF
SET 3,B	CB D8
SET 3,C	CB D9
SET 3,D	CB DA
SET 3,E	CB DB
SET 3,H	CB DC
SET 3,L	CB DD
SET 4,(HL)	CB E6
SET 4,(IX+dis)	DD CB XX E6
SET 4,(IY+dis)	FD CB XX E6
SET 4,A	CB E7
SET 4,B	CB E0
SET 4,C	CB E1
SET 4,D	CB E2
SET 4,E	CB E3
SET 4,H	CB E4
SET 5,(HL)	CB EE
SET 5,(IX+dis)	DD CB XX EE
SET 5,(IY+dis)	FD CB XX EE
SET 5,A	CB EF
SET 5,B	CB E8
SET 5,C	CB E9
SET 5,D	CB EA
SET 5,E	CB EB
SET 5,H	CB EC
SET 5,L	CB ED
SET 6,(HL)	CB F6
SET 6,(IX+dis)	DD CB XX F6
SET 6,(IY+dis)	FD CB XX F6
SET 6,A	CB F7
SET 6,B	CB F0
SET 6,C	CB F1
SET 6,D	CB F2
SET 6,E	CBF3
SET 6,H	CB F4
SET 6,L	CB F5
SET 7,(HL)	CB FE
SET 7,(IX+dis)	DD CB XX FE
SET 7,(IY+dis)	FD CB XX FE
SET 7,A	CB FF
SET 7,B	CB FB
SET 7,C	CB F9
SET 7,D	CB FA
SET 7,E	CB FB
SET 7,H	CB FC
SET 7,L	CB FD
SLA (HL)	CB 26
SLA (IX+dis)	DD CB XX 26
SLA (IY+dis)	FD CB XX 26
SLA A	CB 27
SLA B	CB 20
SLA C	CB 21
SLA D	CB 22
SLA E	CB 23
SLA H	CB 24
SLA L	CB 25
SRA (HL)	CB 2E
SRA (IX+dis)	DD CB XX 2E
SRA (IY+dis)	FD CB XX 2E
SRA A	CB 2F
SRA B	CB 28
SRA C	CB 29
SRA D	CB 2A
SRA E	CB 2B
SRA H	CB 2C
SRA L	CB 2D
SRL (HL)	CB 3E
SRL (IX+dis)	DD CB XX 3E
SRL (IY+dis)	FD CB XX 3E
SRL A	CB 3F
SRL B	CB 38
SRL C	CB 39
SRL D	CB 3A
SRL E	CB 3B
SRL H	CB 3C
SRL L	CB 3D
SUB (HL)	96
SUB (IX+dis)	DD 96 XX
SUB (IY+dis)	FD 96 XX
SUB A	97
SUB B	90
SUB C	91
SUB D	92
SUB dd	D6 dd
SUB E	93
SUB H	94
SUB L	95
XOR (HL)	AE
XOR (IX+dis)	DD AE XX
XOR (IY+dis)	FD AE XX
XOR A	AF
XOR B	AB
XOR C	A9
XOR D	AA
XOR dd	EE dd
XOR E	AB
XOR H	AC
XOR L	AD

SET 4,L

CB E5

corrected 16 - 6 - 1985

Z80 Machine Codes FOR DISASSEMBLY

SHEET 12

This is a Z-80 MACHINE CODE disassembly table. Use it in conjunction with the Z-80 Machine Codes presented previously, for the creation of your own programs.

These lists make programming and disassembly easy. Fit them into a plastic sleeve and keep them handy.

00	NOP
01	LD BC,dddd
02	LD (BC),A
03	INC BC
04	INC B
05	DEC B
06	LD B,dd
07	RLCA
08	EX AF,A'F'
09	ADD HL,BC
0A	LD A,(BC)
0B	DEC BC
0C	INC C
0D	DEC C
0E	LD C,dd
0F	RRCA
10	DJNZ dis
11	LD DE,dddd
12	LD (DE),A
13	INC DE
14	INC D
15	DEC D
16	LD D,dd
17	RLA
18	JR dis
19	ADD HL,DE
1A	LD A,(DE)
1B	DEC DE
1C	INC E
1D	DEC E
1E	LD E,dd
1F	RRA
20	JR NZ dis
21	LD HL,dddd
22	LD (ADDR),HL
23	INC HL
24	INC H
25	DEC H
26	LD H,dd
27	DAA
28	JR Z,dis
29	ADD HL,HL
2A	LD HL,(ADDR)
2B	DEC HL
2C	INC L
2D	DEC L
2E	LD L,dd
2F	CPL
30	JR NC,dis
31	LD SP,dddd
32	LD (ADDR),A
33	INC SP
34	INC (HL)
35	DEC (HL)
36	LD (HL),dd
37	SCF
38	JR C,dis
39	ADD HL,SP
3A	LD A,(ADDR)
3B	DEC SP
3C	INC A
3D	DEC A
3E	LD A,dd
3F	CCF
40	LD B,B
41	LD B,C
42	LD B,D
43	LD B,E
44	LD B,H
45	LD B,L
46	LD B,(HL)
47	LD B,A
48	LD C,B
49	LD C,C
4A	LD C,D
4B	LD C,E
4C	LD C,H
4D	LD C,L
4E	LD C,(HL)
4F	LD C,A
50	LD D,B
51	LD D,C
52	LD D,D
53	LD D,E
54	LD D,H
55	LD D,L
56	LD D,(HL)
57	LD D,A
58	LD E,B
59	LD E,C
5A	LD E,D
5B	LD E,E
5C	LD E,H
5D	LD E,L
5E	LD E,(HL)
5F	LD E,A
60	LD H,B
61	LD H,C
62	LD H,D
63	LD H,E
64	LD H,H
65	LD H,L
66	LD H,(HL)
67	LD H,A
68	LD L,B
69	LD L,C
6A	LD L,D
6B	LD L,E
6C	LD L,H
6D	LD L,L
6E	LD L,(HL)
6F	LD L,A
70	LD (HL),B
71	LD (HL),C
72	LD (HL),D
73	LD (HL),E
74	LD (HL),H
75	LD (HL),L
76	HALT
77	LD (HL),A
78	LD A,B
79	LD A,C
7A	LD A,D
7B	LD A,E
7C	LD A,H
7D	LD A,L
7E	LD A,(HL)
7F	LD A,A
80	ADD A,B
81	ADD A,C
82	ADD A,D
83	ADD A,E
84	ADD A,H
85	ADD A,L
86	ADD A,(HL)
87	ADD A,A
88	ADC A,B
89	ADC A,C
8A	ADC A,D
8B	ADC A,E
8C	ADC A,H
8D	ADC A,L
8E	ADC A,(HL)
8F	ADC A,A
90	SUB B
91	SUB C
92	SUB D
93	SUB E
94	SUB H
95	SUB L
96	SUB (HL)
97	SUB A
98	SBC A,B
99	SBC A,C
9A	SBC A,D
9B	SBC A,E
9C	SBC A,H
9D	SBC A,L
9E	SBC A,(HL)
9F	SBC A,A
A0	AND B
A1	AND C
A2	AND D
A3	AND E
A4	AND H
A5	AND L
A6	AND (HL)
A7	AND A
A8	XOR B
A9	XOR C
AA	XOR D
AB	XOR E
AC	XOR H
AD	XOR L
AE	XOR (HL)
AF	XOR A
B0	OR B
B1	OR C
B2	OR D
B3	OR E
B4	OR H
B5	OR L
B6	OR (HL)
B7	OR A
B8	CP B
B9	CP C
BA	CP D
BB	CP E
BC	CP H
BD	CP L
BE	CP (HL)
BF	CP A
C0	RET NZ
C1	POP BC
C2	JP NZ ADDR
C3	JP ADDR
C4	CALL NZ ADDR
C5	PUSH BC
C6	ADD A,dd
C7	RST 00
C8	RET Z
C9	RET
CA	JP Z ADDR
CB	★
CC	CALL Z,ADDR
CD	CALL ADDR
CE	ADC A,dd
CF	RST 08
D0	RET NC
D1	POP DE
D2	JP NC ADDR
D3	OUT port,A
D4	CALL NC ADDR
D5	PUSH DE
D6	SUB dd
D7	RST 10
D8	RET C
D9	EXX
DA	JP C ADDR
DB	IN A,port
DC	CALL C ADDR
DD	★
DE	SBC A,dd
DF	RST 18
E0	RET PO
E1	POP HL
E2	JP PO ADDR
E3	EX (SP),HL
E4	CALL PO ADDR
E5	PUSH HL
E6	AND dd
E7	RST 20
E8	RET PE
E9	JP (HL)
EA	JP PE ADDR
EB	EX DE,HL
EC	CALL PE ADDR
ED	★
EE	XOR dd
EF	RST 28
F0	RET P
F1	POP AF
F2	JP P ADDR
F3	DI
F4	CALL P ADDR
F5	PUSH AF
F6	OR dd
F7	RST 30
F8	RET M
F9	LD SP,HL
FA	JP M ADDR
FB	EI
FC	CALL M ADDR
FD	★
FE	CP dd
FF	RST 38
CB 00	RLC B
CB 01	RLC C
CB 02	RLC D
CB 03	RLC E
CB 04	RLC H
CB 05	RLC L
CB 06	RLC (HL)
CB 07	RLC A
CB 08	RRC B
CB 09	RRC C
CB 0A	RRC D
CB 0B	RRC E
CB 0C	RRC H
CB 0D	RRC L
CB 0E	RRC (HL)
CB 0F	RRC A
CB 10	RL B
CB 11	RL C
CB 12	RL D
CB 13	RL E
CB 14	RL H
CB 15	RL L
CB 16	RL (HL)
CB 17	RL A
CB 18	RR B
CB 19	RR C
CB 1A	RR D
CB 1B	RR E
CB 1C	RR H
CB 1D	RR L
CB 1E	RR (HL)
CB 1F	RR A
CB 20	SLA B
CB 21	SLA C
CB 22	SLA D
CB 23	SLA E
CB 24	SLA H
CB 25	SLA L
CB 26	SLA (HL)
CB 27	SLA A
CB 28	SRA B
CB 29	SRA C
CB 2A	SRA D
CB 2B	SRA E
CB 2C	SRA H
CB 2D	SRA L
CB 2E	SRA (HL)
CB 2F	SRA A
CB 38	SRL B
CB 39	SRL C
CB 3A	SRL D
CB 3B	SRL E
CB 3C	SRL H
CB 3D	SRL L
CB 3E	SRL (HL)
CB 3F	SRL A
CB 40	BIT 0,B
CB 41	BIT 0,C
CB 42	BIT 0,D
CB 43	BIT 0,E
CB 44	BIT 0,H
CB 45	BIT 0,L
CB 46	BIT 0,(HL)
CB 47	BIT 0,A
CB 48	BIT 1,B
CB 49	BIT 1,C
CB 4A	BIT 1,D
CB 4B	BIT 1,E
CB 4C	BIT 1,H
CB 4D	BIT 1,L
CB 4E	BIT 1,(HL)
CB 4F	BIT 1,A
CB 50	BIT 2,B
CB 51	BIT 2,C
CB 52	BIT 2,D
CB 53	BIT 2,E
CB 54	BIT 2,H
CB 55	BIT 2,L
CB 56	BIT 2,(HL)
CB 57	BIT 2,A
CB 58	BIT 3,B
CB 59	BIT 3,C
CB 5A	BIT 3,D
CB 5B	BIT 3,E
CB 5C	BIT 3,H
CB 5D	BIT 3,L
CB 5E	BIT 3,(HL)
CB 5F	BIT 3,A
CB 60	BIT 4,B
CB 61	BIT 4,C
CB 62	BIT 4,D
CB 63	BIT 4,E
CB 64	BIT 4,H
CB 65	BIT 4,L
CB 66	BIT 4,(HL)
CB 67	BIT 4,A
CB 68	BIT 5,B
CB 69	BIT 5,C
CB 6A	BIT 5,D
CB 6B	BIT 5,E
CB 6C	BIT 5,H
CB 6D	BIT 5,L
CB 6E	BIT 5,(HL)
CB 6F	BIT 5,A
CB 70	BIT 6,B
CB 71	BIT 6,C
CB 72	BIT 6,D
CB 73	BIT 6,E
CB 74	BIT 6,H
CB 75	BIT 6,L
CB 76	BIT 6,(HL)
CB 77	BIT 6,A
CB 78	BIT 7,B
CB 79	BIT 7,C
CB 7A	BIT 7,D
CB 7B	BIT 7,E
CB 7C	BIT 7,H
CB 7D	BIT 7,L
CB 7E	BIT 7,(HL)
CB 7F	BIT 7,A
CB 80	RES 0,B
CB 81	RES 0,C
CB 82	RES 0,D
CB 83	RES 0,E
CB 84	RES 0,H
CB 85	RES 0,L
CB 86	RES 0,(HL)
CB 87	RES 0,A
CB 88	RES 1,B
CB 89	RES 1,C
CB 8A	RES 1,D
CB 8B	RES 1,E
CB 8C	RES 1,H
CB 8D	RES 1,L
CB 8E	RES 1,(HL)
CB 8F	RES 1,A
CB 90	RES 2,B
CB 91	RES 2,C
CB 92	RES 2,D
CB 93	RES 2,E
CB 94	RES 2,H
CB 95	RES 2,L
CB 96	RES 2,(HL)
CB 97	RES 2,A
CB 98	RES 3,B
CB 99	RES 3,C
CB 9A	RES 3,D
CB 9B	RES 3,E
CB 9C	RES 3,H
CB 9D	RES 3,L
CB 9E	RES 3,(HL)
CB 9F	RES 3,A
CB A0	RES 4,B
CB A1	RES 4,C
CB A2	RES 4,D
CB A3	RES 4,E
CB A4	RES 4,H
CB A5	RES 4,L
CB A6	RES 4,(HL)
CB A7	RES 4,A
CB A8	RES 5,B
CB A9	RES 5,C
CB AA	RES 5,D
CB AB	RES 5,E
CB AC	RES 5,H
CB AD	RES 5,L
CB AE	RES 5,(HL)
CB AF	RES 5,A
CB B0	RES 6,B
CB B1	RES 6,C
CB B2	RES 6,D
CB B3	RES 6,E
CB B4	RES 6,H
CB B5	RES 6,L
CB B6	RES 6,(HL)
CB B7	RES 6,A
CB B8	RES 7,B
CB B9	RES 7,C
CB BA	RES 7,D
CB BB	RES 7,E
CB BC	RES 7,H
CB BD	RES 7,L
CB BE	RES 7,(HL)
CB BF	RES 7,A
CB C0	SET 0,B
CB C1	SET 0,C
CB C2	SET 0,D
CB C3	SET 0,E
CB C4	SET 0,H
CB C5	SET 0,L
CB C6	SET 0,(HL)
CB C7	SET 0,A
CB C8	SET 1,B
CB C9	SET 1,C
CB CA	SET 1,D
CB CB	SET 1,E
CB CC	SET 1,H
CB CD	SET 1,L
CB CE	SET 1,(HL)
CB CF	SET 1,A
CB D0	SET 2,B
CB D1	SET 2,C
CB D2	SET 2,D
CB D3	SET 2,E
CB D4	SET 2,H
CB D5	SET 2,L
CB D6	SET 2,(HL)
CB D7	SET 2,A
CB D8	SET 3,B
CB D9	SET 3,C
CB DA	SET 3,D
CB DB	SET 3,E
CB DC	SET 3,H
CB DD	SET 3,L
CB DE	SET 3,(HL)
CB DF	SET 3,A
CB E0	SET 4,B
CB E1	SET 4,C
CB E2	SET 4,D
CB E3	SET 4,E
CB E4	SET 4,H
CB E5	SET 4,L
CB E6	SET 4,(HL)
CB E7	SET 4,A
CB E8	SET 5,B
CB E9	SET 5,C
CB EA	SET 5,D
CB EB	SET 5,E
CB EC	SET 5,H
CB ED	SET 5,L
CB EE	SET 5,(HL)
CB EF	SET 5,A
CB F0	SET 6,B
CB F1	SET 6,C
CB F2	SET 6,D
CB F3	SET 6,E
CB F4	SET 6,H
CB F5	SET 6,L
CB F6	SET 6,(HL)
CB F7	SET 6,A
CB F8	SET 7,B
CB F9	SET 7,C
CB FA	SET 7,D
CB FB	SET 7,E
CB FC	SET 7,H
CB FD	SET 7,L
CB FE	SET 7,(HL)
CB FF	SET 7,A
DD 09	ADD IX,BC
DD 19	ADD IX,DE
DD 21	LD IX,dddd
DD 22	LD (ADDR),IX
DD 23	INC IX
DD 29	ADD IX,IX
DD 2A	LD IX,(ADDR)
DD 2B	DEC IX
DD 34	INC,(IX + dis)
DD 35	DEC,(IX + dis)
DD 36	LD (IX + dis)dd
DD 39	ADD IX,SP
DD 46	LD B,(IX + dis)
DD 4E	LD C,(IX + dis)
DD 56	LD D,(IX + dis)
DD 5E	LD E,(IX + dis)
DD 66	LD H,(IX + dis)
DD 6E	LD L,(IX + dis)
DD 70	LD (IX + dis),B
DD 71	LD (IX + dis),C
DD 72	LD (IX + dis),D
DD 73	LD (IX + dis),E
DD 74	LD (IX + dis),H
DD 75	LD (IX + dis),L
DD 77	LD (IX + dis),A
DD 7E	LD A,(IX + dis)
DD 86	ADD A,(IX + dis)
DD 8E	ADC A,(IX + dis)
DD 96	SUB (IX + dis)
DD 9E	SBC A,(IX + dis)
DD A6	AND (IX + dis)
DD AE	XOR (IX + dis)
DD B6	OR (IX + dis)
DD BE	CP (IX + dis)
DD CB XX 06	RLC (IX + dis)
DD CB XX 0E	RRC (IX + dis)
DD CB XX 16	RL (IX + dis)
DD CB XX 1E	RR (IX + dis)
DD CB XX 26	SLA (IX + dis)
DD CB XX 2E	SRA (IX + dis)
DD CB XX 3E	SRL (IX + dis)
DD CB XX 46	BIT 0,(IX + dis)
DD CB XX 4E	BIT 1,(IX + dis)
DD CB XX 56	BIT 2,(IX + dis)
DD CB XX 5E	BIT 3,(IX + dis)
DD CB XX 66	BIT 4,(IX + dis)
DD CB XX 6E	BIT 5,(IX + dis)
DD CB XX 76	BIT 6,(IX + dis)
DD CB XX 7E	BIT 7,(IX + dis)
DD CB XX 86	RES 0,(IX + dis)
DD CB XX 8E	RES 1,(IX + dis)
DD CB XX 96	RES 2,(IX + dis)
DD CB XX 9E	RES 3,(IX + dis)
DD CB XX A6	RES 4,(IX + dis)
DD CB XX AE	RES 5,(IX + dis)
DD CB XX B6	RES 6,(IX + dis)
DD CB XX BE	RES 7,(IX + dis)
DD CB XX C6	SET 0,(IX + dis)
DD CB XX CE	SET 1,(IX + dis)
DD CB XX D6	SET 2,(IX + dis)
DD CB XX DE	SET 3,(IX + dis)
DD CB XX E6	SET 4,(IX + dis)
DD CB XX EE	SET 5,(IX + dis)
DD CB XX F6	SET 6,(IX + dis)
DD CB XX FE	SET 7,(IX + dis)
DD E1	POP IX
DD E3	EX (SP),IX
DD E5	PUSH IX
DD E9	JP (IX)
DD F9	LD SP,IX
ED 40	IN B,(C)
ED 41	OUT (C),B
ED 42	SBC HL,BC
ED 43	LD (ADDR),BC
ED 44	NEG
ED 45	RETN
ED 46	IM 0
ED 47	LD I,A
ED 48	IN C,(C)
ED 49	OUT (C),C
ED 4A	ADC HL,BC
ED 4B	LD BC,(ADDR)
ED 4D	RETI
ED 4F	LD R,A
ED 50	IN D,(C)
ED 51	OUT (C),D
ED 52	SBC HL,DE
ED 53	LD (ADDR),DE
ED 56	IM 1
ED 57	LD A,I
ED 58	IN E,(C)
ED 59	OUT (C),E
ED 5A	ADC HL,DE
ED 5B	LD DE,(ADDR)
ED 5E	IM 2
ED 5F	LD A,R
ED 60	IN H,(C)
ED 61	OUT (C),H
ED 62	SBC HL,HL
ED 63	LD (ADDR),HL
ED 67	RRD
ED 68	IN L,(C)
ED 69	OUT (C),L
ED 6A	ADC HL,HL
ED 6B	LD HL,(ADDR)
ED 6F	RLD
ED 72	SBC HL,SP
ED 73	LD (ADDR),SP
ED 78	IN A,(C)
ED 79	OUT (C),A
ED 7A	ADC HL,SP
ED 7B	LD SP,(ADDR)
ED A0	LDI
ED A1	CPI
ED A2	INI
ED A3	OUTI
ED A8	LDD
ED A9	CPD
ED AA	IND
ED AB	OUTD
ED B0	LDIR
ED B1	CPIR
ED B2	INIR
ED B3	DTIR
ED B8	LDDR
ED B9	CPDR
ED BA	INDR
ED BB	OTDR
FD 09	ADD IY,BC
FD 19	ADD IY,DE
FD 21	LD IY,dddd
FD 22	LD (ADDR),IY
FD 23	INC IY
FD 29	ADD IY,IY
FD 2A	LD IY,(ADDR)
FD 2B	DEC IY
FD 34	INC,(IY + dis)
FD 35	DEC,(IY + dis)
FD 36	LD (IY + dis)dd
FD 39	ADD IY,SP
FD 46	LD B,(IY + dis)
FD 4E	LD C,(IY + dis)
FD 56	LD D,(IY + dis)
FD 5E	LD E,(IY + dis)
FD 66	LD H,(IY + dis)
FD 6E	LD L,(IY + dis)
FD 70	LD (IY + dis),B
FD 71	LD (IY + dis),C
FD 72	LD (IY + dis),D
FD 73	LD (IY + dis),E
FD 74	LD (IY + dis),H
FD 75	LD (IY + dis),L
FD 77	LD (IY + dis),A
FD 7E	LD A,(IY + dis)
FD 86	ADD A,(IY + dis)
FD 8E	ADC A,(IY + dis)
FD 96	SUB (IY + dis)
FD 9E	SBC A,(IY + dis)
FD A6	AND (IY + dis)
FD AE	XOR (IY + dis)
FD B6	OR (IY + dis)
FD BE	CP (IY + dis)
FD CB XX 06	RLC (IY + dis)
FD CB XX 0E	RRC (IY + dis)
FD CB XX 16	RL (IY + dis)
FD CB XX 1E	RR (IY + dis)
FD CB XX 26	SLA (IY + dis)
FD CB XX 2E	SRA (IY + dis)
FD CB XX 3E	SRL (IY + dis)
FD CB XX 46	BIT 0,(IY + dis)
FD CB XX 4E	BIT 1,(IY + dis)
FD CB XX 56	BIT 2,(IY + dis)
FD CB XX 5E	BIT 3,(IY + dis)
FD CB XX 66	BIT 4,(IY + dis)
FD CB XX 6E	BIT 5,(IY + dis)
FD CB XX 76	BIT 6,(IY + dis)
FD CB XX 7E	BIT 7,(IY + dis)
FD CB XX 86	RES 0,(IY + dis)
FD CB XX 8E	RES 1,(IY + dis)
FD CB XX 96	RES 2,(IY + dis)
FD CB XX 9E	RES 3,(IY + dis)
FD CB XX A6	RES 4,(IY + dis)
FD CB XX AE	RES 5,(IY + dis)
FD CB XX B6	RES 6,(IY + dis)
FD CB XX BE	RES 7,(IY + dis)
FD CB XX C6	SET 0,(IY + dis)
FD CB XX CE	SET 1,(IY + dis)
FD CB XX D6	SET 2,(IY + dis)
FD CB XX DE	SET 3,(IY + dis)
FD CB XX E6	SET 4,(IY + dis)
FD CB XX EE	SET 5,(IY + dis)
FD CB XX F6	SET 6,(IY + dis)
FD CB XX FE	SET 7,(IY + dis)
FD E1	POP IY
FD E3	EX (SP),IY
FD E5	PUSH IY
FD E9	JP (IY)
FD F9	LD SP,IY

Z80 Machine Codes EXPLAINED Part 1

SHEET 13

A **BRIEF** description for each Z-80 instruction:

ADC A,( ) . The value of the byte pointed to by the address in ( ) **plus the value of the carry flag**, is added to the accumulator.

ADC A,B . . The value of the B register **plus the value of the carry flag** is added to the accumulator.

ADC HL,BC . The value of the register pair BC **plus the value of the carry flag** , is added to the HL register pair.

ADD A,( ) . The byte at the address ( ) is added to the accumulator.

ADD A,B . . The value of the B register is added to the accumulator.

ADD HL,BC . The value of the BC register pair is added to the HL register pair.

ADD IX,BC . The value of the BC register pair is added to the index register.

AND ( ) . . The value of the byte pointed to by the address in ( ) is logically **AND**ed with the accumulator.

AND A . . . The accumulator is **AND**ed with itself.

AND B . . . The B register is **AND**ed with the accumulator.

BIT 0,( ) . . Bit 0 of the byte pointed to by the address in ( ) is tested and if found to be '0', the **ZERO FLAG** is set to '1'.

BIT 0,A . . . Bit 0 of the A register is tested and if it is '1' the zero flag is set to '0'.

CALL Addr . . The program is diverted to a sub-routine.

CALL C Address The CALL will only be performed if the carry flag in the F register is '1'.

CALL M Address The CALL will only be performed if the S flag (sign flag) is negative.

CALL NC Address The CALL will only be performed if the NON-CARRY condition is present. i.e. carry flag '0'.

CALL NZ Address The CALL will only be performed if a **NON-ZERO** condition is satisfied. i.e. the zero flag is '0'.

CALL P Address The CALL will only be performed if the sign flag in the F register is positive. i.e S = 1.

CALL PE Address The CALL instruction will only be executed if the PARITY is EVEN. This means the P/V flag is SET (1).

CALL PO Address The CALL directive will only be executed if the PARITY is ODD. This means the P/V flag is reset (0).

CALL Z Address The CALL will only be executed if the zero flag is SET (1).

CCF . . . . . Complements the CARRY FLAG - reverses the condition of the carry flag.

CP ( ) . . . Compare the value of the byte pointed to by the address in ( ) with the accumulator.

CP A . . . . The accumulator is compared with itself.

CP B . . . . The value of the B register is compared with the accumulator.

CPD . . . . The contents of the memory location pointed to by the HL register pair is subtracted from the accumulator and the result discarded. Both HL and BC are decremented.

CPDR . . . . As above but instruction will terminate when BC = 0 or A = (HL).

CPI . . . . . The contents of the memory location pointed to by the HL register pair are compared with the contents of the accumulator. HL is then incremented and BC decremented.

CPIR . . . . Compare the contents of the address pointed to by the HL register pair with the accumulator. HL is then incremented and BC decremented until BC = 0. or if A = (HL).

CPL . . . . . Complement the accumulator. i.e. all 1's are changed to 0's etc.

DAA . . . . . Decimal Adjust the Accumulator. Produces one digit for the 4 least significant bits and one for the 4 most significant bits. The carry flag is set to '1' if an overflow occurs.

DEC ( ) . . . Decrement the contents of a memory location by one.

DEC A . . . . Decrement the contents of a CPU register by one.

DI . . . . . Disable a maskable interrupt signal.

DJNZ . . . . A conditional relative addressing jump. The contents of register B is firstly decremented. If the result is NOT ZERO, a jump, determined by the value of the displacement byte, will take place. If the result is zero, the next instruction will be executed.

EI . . . . . This one-byte instruction ENABLES the maskable interrupt function by setting the interrupt flip flops.

EX (SP)HL . The contents of the location addressed by the Stack Pointer are exchanged with the contents of the CPU register L. The contents of the H register are exchanged with the contents of the stack pointer plus one.

EX AF,AF' . . The contents of the accumulator and status register are exchanged with the contents of the alternate accumulator and status register.

EX DE,HL . . Exchange the contents of DE and HL registers.

EXX . . . . Exchange the contents of the general purpose registers with corresponding alternate registers.

HALT . . . . CPU suspends operation and executes NOP's. It maintains memory refresh logic.

IM 0 . . . . Sets interrupt mode 0.

IM 1 . . . . Program RESTARTS at 0038H when interrupted.

IM 2 . . . . Sets interrupt mode 2.

IN A,(C) . . Data (from the input port specified by the contents of register C) is loaded into register A.

INC (HL) . . The contents of a memory location are incremented by one.

INC A . HL Increment register A (Or a register Pair e.g. BC DE etc.)

IND . . . . Input from a port specified by the contents of register C. One byte of data is transferred to the memory location addressed by the contents of the HL register pair. The value in register B and HL will be decremented at the end of this instruction.

INDR . . . . Same as IND except the instruction continues until register B reaches zero.

INI . . . . . Same as IND except the contents of HL register pair are decremented at the conclusion of the instruction.

INIR . . . . Same as INI except the instruction repeats until register B reaches zero.

JP (HL) . . . Jump to the address contained in register pair HL.

JP ADDR . . See CALL instructions.

JP C,ADDR . A conditional relative addressing jump. See CALL for

JP M,ADDR . meaning of C, NC, NZ & Z.

LD (ADDR),A The contents of the accumulator are loaded into the address contained within the ( ).

LD (ADDR)BC The contents of C are loaded into the immediate address and B into the address plus one.

LD (BC),A . . The contents of the accumulator are loaded into the location pointed to by the contents of BC.

LD ( ),A (IX + dis) . . The contents of A are loaded into the memory location obtained by adding a displacement value to the contents of the IX register.

LD A,(ADDR) Load the accumulator with the contents of the immediately specified memory address.

LD A,A . . . Load the data from one CPU register into another.

LD A,dd . . Load one byte of data into the CPU register specified.

LD BC,dd dd Load 2 bytes of data into the CPU register pair specified. e.g. The first byte loads into C and the second byte into B.

LD A,I . . . Load the accumulator with the contents of the Interrupt Vector register.

LD A,R . . . Load the accumulator with the contents of the Memory Refresh register.

LD I,A . . . Load the Interrupt vector register, from the accumulator.

LD R,A . . . Load the Memory Refresh register with the contents of the accumulator.

LDD . . . . The contents of a memory location pointed to by the contents of the HL register pair are transferred to the location pointed to by the contents of the DE register pair. After the data has been transferred both HL and DE are decremented by one. Also the 'counter-register' pair BC is decreased by one.

LDDR . . . . Same as LDD except if contents of BC do not go to zero, the Program Counter will be decreased by a value of two and the instruction will be re-executed. The instruction will continue until the value in the register pair BC goes to zero.

LDI . . . . . Same as LDD except register pairs HL and DE are increased by a count of one.

LDIR . . . . Same as LDDR except register pairs HL and DE are incremented by one after the data has been transferred.

Z80 Machine Codes EXPLAINED Part 11

NEG . . . . . Each bit in the accumulator is reversed sign. One's go to zero and zero's go to one. Then one is added to the result.

NOP . . . . . The NO OPERATION instruction. Only the Program Counter advances.

OR ( ) . . . The logic OR operation is performed between the accumulator and the contents of the memory location pointed to by the address in ( ).

OR A,B,C etc A logic OR operation is performed between the accumulator and a specified register.

OTDR . . . . Data from the memory location specified by the contents of the HL register is outputted to a port as specified by the contents of register C. The HL pointer has its value decremented after each transfer operation. The value of register B is decremented and if the result is zero, the Program Counter register is set back 2 units so that the instruction is re-executed.

OTIR . . . . Same OTDR except that the HL pointer is incremented after each execution.

OUT (C),A etc The contents of A, B, C, D, etc are outputted to the port specified by the contents of register C.

OUT port,A . . The contents of the accumulator is outputted to the port specified.

OUTD . . . . Data is outputted from memory location specified by the contents of the HL register pair to the port specified by the contents of register C. (contents of register B will be decremented but no repeat operation will be performed.) HL register pair has its contents decremented after the conclusion of the operation.

OUTI . . . . . Same as OUTD except HL has its contents incremented at the conclusion of the operation.

POP AF . . . Two bytes are removed from the stack. The first byte is loaded into F and the second into A.

PUSH HL . . . Two bytes are placed onto the stack. The contents of the HIGH ORDER register are stored in the stack at the address of the stack pointer less one. The content of the LOW ORDER register are stored at the address of the stack pointer less two.

RES D,( ) . . RESET BIT 0, 1, 2, 3, 4, 5, 6, 7 to the logic ZERO conditon of the specified register.

RET . . . . . The unconditional RETURN instruction

RET C . . . . Return from the sub-routine if the carry flag in the F register is true (1).

RET M . . . . The instruction will only be performed if the S flag (sign flag) is negative.

RET NC . . . The instruction will only be performed if the NON-CARRY condition is present. i.e. the CARRY FLAG is '0'.

RET NZ . . . The instruction will only be performed if a NON-ZERO condition is satisfied. i.e. the ZERO FLAG is '0'.

RET P . . . . The instruction will only be performed if the sign flag in the F register is positive. i.e. S = 1.

RET PE . . . The instruction will only be performed if the PARITY is EVEN. This means the P/V flag is SET (1).

RET PO . . . The instruction will only performed if the PARITY is ODD. This means the P/V flag is reset (0).

RET Z . . . . The instruction will only be performed if the ZERO flag is SET (1).

RETI . . . . . Return from INTERRUPT.

RETN . . . . Return from non-maskable INTERRUPT.

RL ( ) . . . The content of the memory location contained in ( ) is rotated to the left, through the carry bit.

RL A,B,C,etc . The contents of the register is rotated one bit position to the left, through the carry bit.

RLA . . . . . This is a one-byte instruction of **RL A** and rotates the contents of the accumulator one bit position to the left, through the carry bit.

RLC ( ) . . . The contents of a memory location pointed to by the contents of the location in ( ) is shifted one bit to the left but not through the carry. The C flag is set to the original status of the register's least significant bit.

RLC A,B,etc . The contents of the indicated register is shifted one bit position to the left. It does not shift through the carry bit but does set the C flag to the original status of the register's most significant bit.

RLCA . . . . This is a one byte instruction of **RLC A** and operates as above.

RLD . . . . The 4 low-order bits of a memory location (pointed to by the contents of register pair in brackets) are transferred to the 4 high-order bits of the same memory location. The 4 high-order bits are transferred into the 4 low-order bits of the accumulator. The previous 4 low-order bits of the accumulator are transferred to the 4 low-order bits of the memory location specified above.

RR ( ) . . . Rotate the contents of a memory location pointed to by the contents of the register in ( ) to the right, through the carry bit.

RR A,B,C etc Rotate the indicated register to the right through the carry bit.

RRA . . . . Rotate the accumulator right, through the carry bit.

RRC ( ) . . Rotate the contents of a memory location pointed to by the contents of the register in ( ) to the right but not through the carry bit. The C flag is set to the status of the register's least significant bit.

RRC A,B,etc Rotate register to the right but not through the carry bit.

RRCA . . . . A one-byte instruction for RRC A.

RRD . . . . The 4 high-order bits of a memory location (pointed to by the contents of register pair HL) are transferred to the 4 low-order bits of the same location. The 4 low-order bits are transferred to the 4 low-order bits of the accumulator. The previous 4 low-order bits of the accumulator are transferred to the 4 high-order bits of the memory location specified above. A special one-

RST 00 . . . A special one-byte subroutine call directive called RESTART.
RST 00 will restart at page zero, location 00. i.e. 00 00
RST 08 will restart at location 08. i.e. 00 08 etc.

SBC A,( ) . Subtract the contents of memory pointed to by the contents of the register pair in ( ) and the carry flag from the accumulator. Store the result in the accumulator.

SBC A,B . . The contents of the B register and the carry flag (the C flag in the F register) are subtracted from the contents of the accumulator. The result is stored in the accumulator.

SCF . . . . The carry flag (the C flag in the F register) is set to 1.

SET 0,( ) . Bit 0, etc at the memory location pointed to by the contents of the register in ( ) is set to 1.

SET 0,B . . The indicated bit in the selected register is set to 1.

SLA ( ) . . SHIFT LEFT ARITHMETIC. Shift the contents of the memory location pointed to by the contents of the register in ( ) one bit to the left, resettng the least significant bit to 0.

SLA A,B,etc Shift the contents of the specified register left one bit, resetting the least significant bit to 0.

SRA ( ) . . SHIFT RIGHT ARITHMETIC. Shift the contents of a memory location pointed to by the contents of the register in ( ) to the right. The high-order bit is not altered. Bit 0 is shifted into the carry bit.

SRA A,B,etc Shift the contents of a register one bit to the right. High-order bit is unchanged. Bit 0 is shifted into the carry bit.

SRL ( ) . . SHIFT RIGHT LOGICAL. The contents of the memory location pointed to by the contents of the register in ( ) are shifted to the right. Bit 7 is reset to 0. Bit 0 is shifted into the carry bit.

SRL A,B,etc The contents of the indicated register is shifted one bit position to the right. Bit 7 is reset to 0. Bit 0 is shifted into the carry bit.

SUB ( ) . . Subtract the contents of the memory location pointed to by the contents of the register in ( ) from the accumulator.

SUB A,B,etc Subtract the contents of the specified register from the accumulator.

SUB dd . . . Subtract immediate data from the accumulator.

XOR ( ) . . Perform the Exclusive-OR operation on data pointed to by the contents of the register in ( ) with the accumulator.

XOR A,B,etc Exclusive-OR the contents of the specified register with the accumulator.

XOR dd . . . Exclusive-OR the immediate data with the accumulator.