Ukázka částí kódu.

Zdeněk Pavlů

31.05.2016

Tyto dokumenty jsou zhruba 12 roků staré, přepisuji je z mého pracovního sešitu. Je mi líto tyto části kódu dát do sběru. Jedná se o mé poznámky, některé jsou z dokumentace Intel a jiných materiálů.

```
*************************************************************************
```

Nuluj zonu 1 - 256 byte

```
;vstup          B  -> 1-256 byte  B=0 pak 256 byte
;               HL adr začátku zony
;výstup         B= 0  HL za konec zóny

Nulz        MVI   M,0
            INX   H
            DCR   B
            JNZ   Nulz
*************************************************************************
```

BC<->DE

```
            PUSH  B
            PUSH  D
            POP   B
            POP   D
*************************************************************************
```

A -> B F->C

```
            PUSH  PSW
            POP   B
*************************************************************************
```

DE <-> {SP} záměna DE s vrcholem zásobníku

```
            XCHG
            XTHL
            XCHG
*************************************************************************
```

BC <-> HL

```
            PUSH  B
            XTHL
            POP   B
*************************************************************************
```

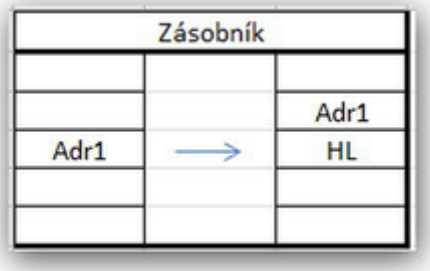
BC<->{SP}

```
            PUSH B
            XTHL
            POP   B     ;záměna BC<->HL
            XTHL
            PUSH  B
            XTHL
            POP   B     ;opětná záměna BC<->HL
*************************************************************************
```

HL -> NOS {SP} -> {SP}

;Obsah HL je uložen do zásobníku na místo současného vrcholu a obsah ;Adr1 nový vrchol.Současného vrcholu o jednu úroveň výše na úroveň nového

```
      XTHL
      PUSH  H
```



****

```
XTHL
PUSH  D
PUSH  B
PUSH  H
```

****

Naplní registry BC,DE,HL ze zásobníku

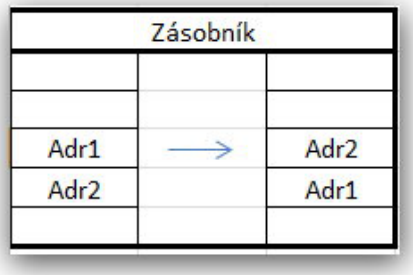
```
POP   H
POP   B
POP   D
XTHL
```

vrchol -> Adr1

****

Výměna posledních položek na vrcholu zásobníku (ničí HL)

```
POP   H
XTHL
PUSH  H
```



****

Uchová všechny registry (riziková operace se zásobníkem nepoužívat při přerušení nebo při blokování přerušení)

```
XTHL
INX         SP
INX         SP
XTHL
DCX         SP
DCX         SP
XTHL
```

****

Přičte obsah A k zadané adrese

```
Vstup reg A – přírustek adresy
      1 a 2 byte za CALL  - počáteční adresa
Výstup HL – vypočtená adresa , uchová BC

Adr1  POP   H
      ADD   M
      MOV   E,A
      INX   H
      MVI   A,0
      ADC   M
      MOV   D,A
      INX   H
      PUSH  H           skutečná návratová adresa v HL
      XCHG
      RET

Adr2  POP   H
      PUSH  D
      ADD   M           TOS skutečná návratová adresa
      MOV   E,A         HL vypočtená adresa
      INX   H           uchovává BC a DE
      MVI   A,0
      ADC   M
      MOV   D,A
      INX   H
      XTHL
      XCHG
      RET
```

Volání podprogramu, nesmí být použito v podmíněném volání. Návratová adr je zvětšena o 2byte, aby pokračování bylo za parametrem

```
CALL  Adr1              CALL Adr2
      DW Pocad          DW   Pocad
```

Pocad + reg A = HL

****

Přičte obsah místa k zadané adrese a uloží do A obsah místa.

```
Vstup:
      Reg A – přírustek adresy
      1 a 2 byte za CALL  - počáteční adresa
Výstup:
      Reg A – obsah vypočteného místa, uchová BC,DE,HL


      CALL  Adr3
      DW    Pocad

Adr3  XTHL
      PUSH  D
      ADD   M
      MOV   E,A
      INX   H
```

```
      MVI   A,0
      ADC   M
      MOV   D,A         VYPOČTENÁ ADRESA
      INX   H
      LDAX  D           reg A - výstup
      POP   D           vráceno DE
      XTHL              vráceno HL
      RET               TOS – návratová adresa
```

Proveď skok na vypočtenou adresu.

```
Vstup:
      Reg A – přírustek adresy
      1 a 2 byte za CALL  - počáteční adresa
Výstup:
      Viz aktivované podprogramy, v okamžiku přechodu uchovává BC,DE,HL

      CALL  Adr4
      DW    Pocad

Adr4  XTHL        TOS0- původní HL
      PUSH  D     TOS1- původní DE
      ADD   M
      MOV   E,A
      INX   H
      MVI   A,0
      ADC   M
      MOV   D,A   DE –vypočtená adresa
      INX   H     HL –návratová adresa
      DI
      INX   SP
      INX   SP    {TOP0}
      XTHL        TOP0 – Návratová adresa vrácena HL
      DCX   SP
      DCX   SP    {TOP1}
      EI
      XCHG
      XTHL        TOP1 – vypočtená adresa
      XCHG        vráceno DE
      RET
```

Přesun slova z určité vrstvy zásobníku.

```
Vstup:
      Reg A – hloubka ponoru, počet úrovní o které je se třeba vrátit
      zpět před okamžikem volání A = vrchol zásobníku.
Výstup:
      HL – výstupní slovo

PON1  MVI   H,0   HL – počet úrovní
      MOV   L,A   zpět o úroveň CALL
      INX   H
      DAD   H     HL ← 2 + HL
      DAD   SP    HL – vypočtená úroveň
      PUSH  D
      MOV   E,M
```

```
      INX   H
      MOV   D,M
      XCHG        HL - výsledek
      POP   D
      RET
```

Když A<>0 pak A=0 když A=0 pak A=FF

```
      SUI   1
      SBB   A
```

Když B<C pak A=FF jinak A=0

```
      MOV   A,B
      SUB   C
      SBB   A
```

Když A>B pak nic když A<B pak A ← B

```
X1    CMP   B
      JNZ   x2
      MOV   A,B
X2    pokračuje
```

Když B=C pak A=FF jinak A=0

```
      MOV   A,B
      SUB   C
      SUI   1           CY=1   B=C
      SBB   A
```

Když B<=C pak A=FF jinak A=0

```
      MOV   A,B
      SUB   C
      MVI   A,FF
      JNZ   PC+7
      JC    PC+4
      XRA   A
```

Když B>C pak A=0 jinak A=FF

```
      MOV   A,C
      SUB   B
      SBB   A
      CMA
```

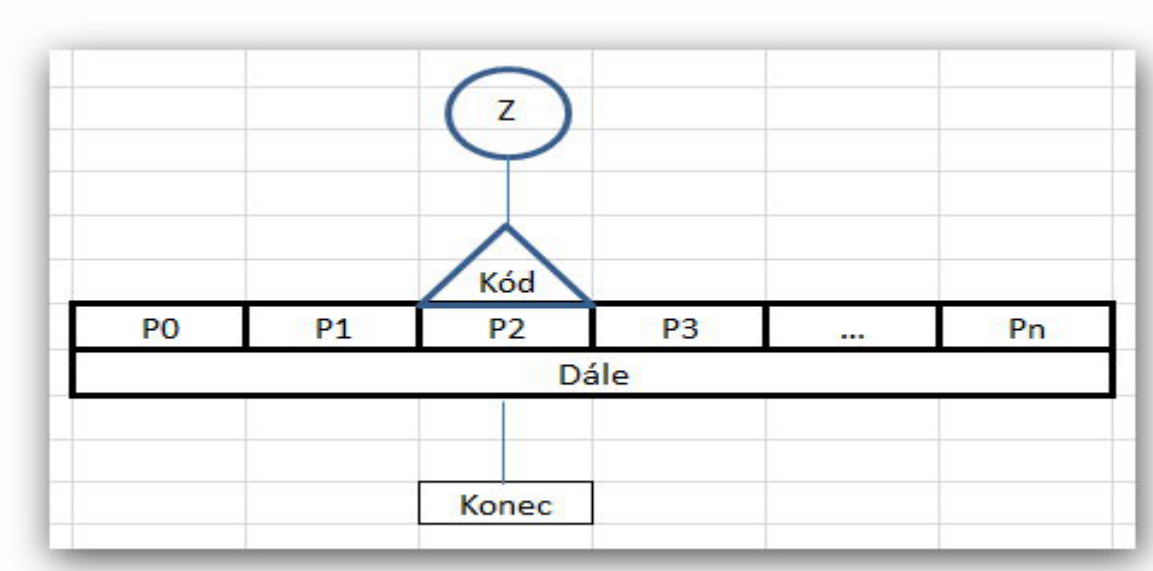
Když A parita B je stejná pak CY=0 jinak CY=1

```
      XRA   B
      JPE   X1
      CMP
X1    POKRAČUJI
```

Když B=ASCI(0-9) pak převede na BCD a dá do A a CY←0 jinak A←0 a CY←1

```
        MOV   A,B
        ANI   7FH
        SUI   30H
        JC    PC+6
        CPI   10
        JNZ   X2
        MVI   A,0
X2      POKRAČUJE
```

Goto:



GOTO1:
Pracuje s tabulkou adres dvou bytovou, předpokládá uspořádanou řadu kódů začínajících 0, nekontroluje překročení ADR tabulky.

GOTO2:
Pracuje s tabulkou relativních adres, jedno bytovou, vzdálenost mezi nejnižší a nejvyšší ADR nesmí překročit 255 byte. Předpokládá uspořádanou řadu kódů začínajících 0 a nekontroluje překročení tabulky.

Vstup:
Registr A vstupní kód
Registr HL začátek tabulky adresa položky 0
ADR tabulky TABAD

```
Goto1   PUSH   D
        XCHG
        MOV    L,A
        MVI    H,0
        DAD    H          HL ADR od počátku tabulky
        DAD    D          HL ADR položky v tabulce
        MOV    E,M
        INX    H
```

```
       MOV   D,M
       XCHG                ADR skoku
       POP   D
       PCHL

TABAD        DW    P0,P1,P2,.....PN
```

K = počet položek 13+(k+1)*2 byte

Vstup:
Registr A vstupní kód
Registr HL začátek tabulky
Registr DE nejnižší adresa skoku

```
Goto2 PUSH  D
      MOV   E,A
      MVI   D,0
      DAD   D           HL ADR položky
      MOV   E,M         DE položka
      POP   H           HL nejnižší ADR
      DAD   D           HL ADR skoku
      PCHL

TABAD       DB P0-P3,P1-P3,P2-P3,0,P4-P3   nejnižší ADR = P3
```

Omezovač tabulky.

```
Vstup B – horní mez

X1    INR   B
      CPI   B
      CMP
      JNC   PC+5        mez je překročena (x2)
      MOV   A,B         ano
X2    MOV   A,B
      DCR   A           A = horní mez, pokud překročena CY=1
```

****

Omezovač tabulky.

```
Vstup B – horní mez
Vstup C – dolní mez

X1    INR   B
      CPI   B
      CMC
      JNZ   PC+8        nad horní?
      MOV   A,B         ano
      DCR   A
      JMP   DALE
      CPI   C
      JNC   DÁLE        pod dolní?
      MOV   A,C         ano
DÁLE  pokračuji
```

Vynuluje CY – neovlivní nic jiného

```
STC
CMC
```

HL + A = HL

```
ADD   L
MOV   L,A
MVI   A,0
ADC   H
MOV   H,A
```

NEBO

```
MOV   E,A
MVI   D,0
DAD   H
```

Budu pokračovat