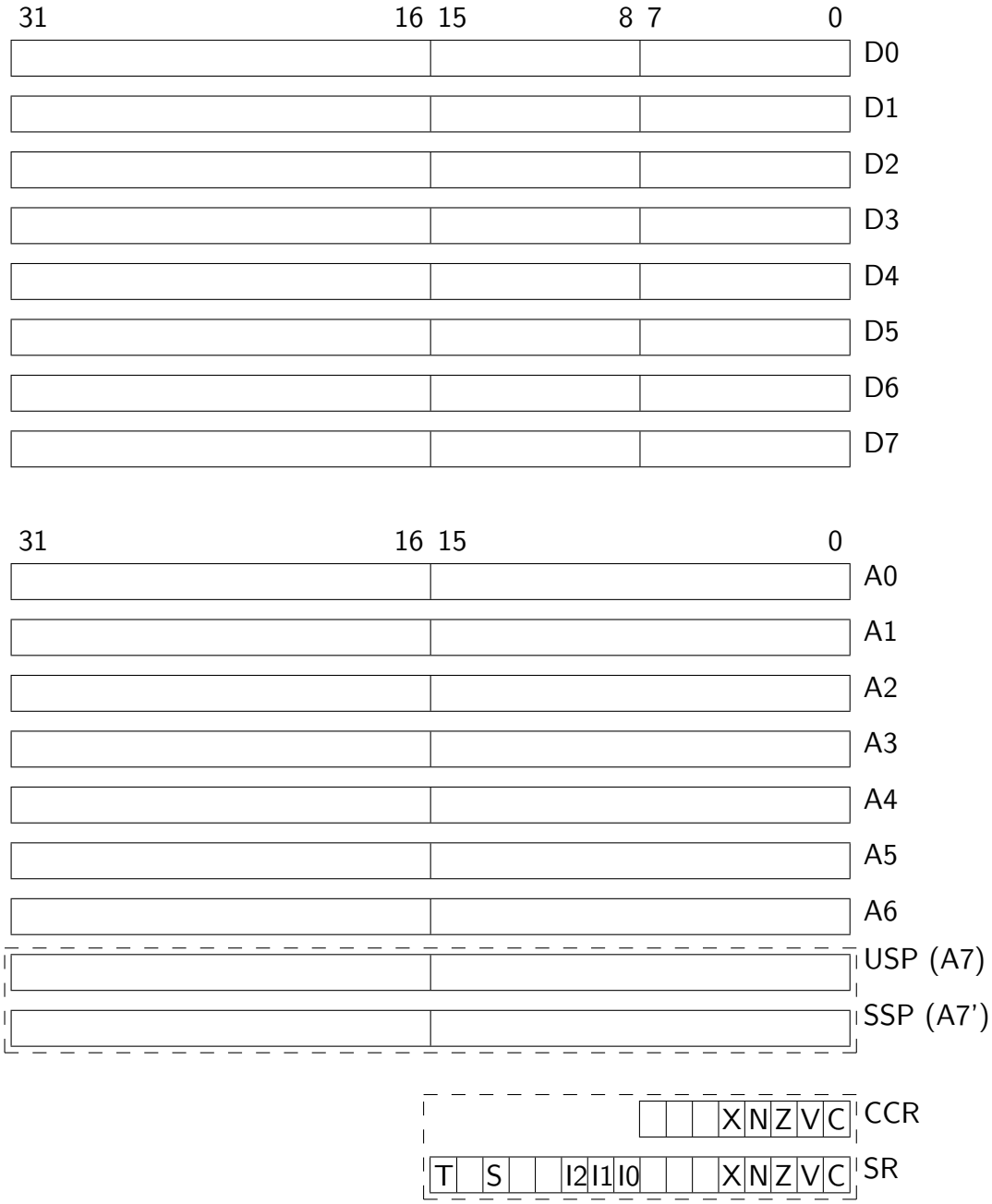
Rejestry procesora



Reprezentacja instrukcji (1/2)

- bezadresowe (*implicit address opcodes*)
- jednoadresowe (*single-address opcodes*)
- półtoraadresowe (*one-and-a-half-address opcodes*)
- dwuadresowe (*two-address opcodes*)

Przykłady:

- `RTS` (ReTurn from Subroutine) - powrót z przerwania
- `JMP <ea>` (JuMP) - skok do adresu
- `ADD <ea>,Dn` - dodanie zawartości adresu do Dn
- `MOVE <ea>,<ea>` - przesłanie zawartości jednego adresu pod drugi

Reprezentacja instrukcji (2/2)

```
RTS                         ((SP)) -> PC

JMP     <ea>                <ea> -> PC

ADD     <ea>,Dn             (<ea>) + (Dn) -> Dn
ADD     Dn,<ea>             (Dn) + (<ea>) -> <ea>

MOVE    <eas>,<ead>         (<eas>) -> <ead>
```

Słowo rozkazowe:

kod operacji, specyfikacja rozmiaru danych, deskryptory adresu (0–2):

```
RTS                 0   % 0 1 0 0 1 1 1 0 0 1 1 1 0 1 0 1
JMP     <ea>        1   % 0 1 0 0 1 1 1 0 1 1 <---ea---->
ADD.B   Dn,<ea>     1.5 % 1 1 0 1 <-Dn> 1 0 0 <---ea---->
ADD.W   Dn,<ea>     1.5 % 1 1 0 1 <-Dn> 1 0 1 <---ea---->
ADD.L   Dn,<ea>     1.5 % 1 1 0 1 <-Dn> 1 1 0 <---ea---->
MOVE.B  <eas>,<ead> 2   % 0 0 0 1 <---ead---> <---eas--->
MOVE.W  <eas>,<ead> 2   % 0 0 1 1 <---ead---> <---eas--->
MOVE.L  <eas>,<ead> 2   % 0 0 1 0 <---ead---> <---eas--->
```

Adresowanie

Przykładowa instrukcja:

```
MOVE.W  <skad>,<dokad>
```

przesyła słowo dwubajtowe (`.W`) z miejsca określonego deskryptorem `skad` do miejsca określonego przez `dokad`.

```
bit:   15 14  13 12  11 10  9  8  7  6   5  4  3  2  1  0
        0  0   1  1  <--------------->  <--------------->
       kod    rozmiar  przeznaczenie          zrodlo
       oper.  danych (dokad-destination)  (skad-source)
```

Deskryptor adresu (*Effective Address*)

```
bit:            5   4   3   2   1   0
               <---------> <--------->
                 rejestr    kod trybu
```

Tryb bezpośredni rejestru danych (*data register direct*)

$$EA = Dn$$

Przykład:

```
      MOVE.W  D1,D2                          (D1) -> D2

  3     4    0    1
<---> <---><---><--->
00 11 010 000 000 001
-- -- --- --- --- ---
 |  |  |   |   |   |_rejestr zrodlowy nr 1      \
 |  |  |   |   |                                 > skad
 |  |  |   |   |_____tryb bezp. rejestru danych /
 |  |  |   |
 |  |  |   |_________tryb bezp. rejestru danych \
 |  |  |                                         > dokad
 |  |  |_____________rejestr przeznaczenia nr 2 /
 |  |
 |  |_________________rozmiar danych - slowo (.W)
 |
 |____________________kod instrukcji MOVE
```

Tryb bezpośredni rejestru adresowego (*address register direct*)

$$EA = An$$

Przykład:

```
      MOVE.W  D1,A2                          (D1) -> A2

00 11 010 001 000 001
-- -- --- --- --- ---
 |  |  |   |   |   |_rejestr zrodlowy nr 1          \
 |  |  |   |   |                                     > skad
 |  |  |   |   |_____tryb bezp. rejestru danych     /
 |  |  |   |
 |  |  |   |__________tryb bezp. rejestru adresowego \
 |  |  |                                             > dokad
 |  |  |______________rejestr przeznaczenia nr 2     /
 |  |
 |  |__________________rozmiar danych - slowo (.W)
 |
 |______________________kod instrukcji MOVE
```

Tryb bezwzględny długi (*absolute long*)

$$EA = ((PC) + 2) : ((PC) + 4)$$

Przykład:

```
DANEWEJ EQU     $123456

        MOVE.W  DANEWEJ,D2          (DANEWEJ) -> D2
```

Kod wytworzony przez asembler – 3 słowa, 4 cykle:

```
  $3435       kod maszynowy MOVE.W z odpowiednimi trybami
  $0012       starsze slowo dlugiego adresu bezwzglednego
  $3456       mlodsze slowo dlugiego adresu bezwzglednego
```

Oba argumenty w pamięci – 5 słów, 6 cykli:

```
DANEWEJ EQU     $123456
DANEWYJ EQU     $345678

        MOVE.W  DANEWEJ,DANEWYJ   (DANEWEJ) -> DANEWYJ
```

Tryb bezwzględny krótki (*absolute short*)

$$EA = ((PC) + 2)_{SEX}$$

Kodowanie adresu:

pole trybu: %111, pole rejestru: %000

Uwaga:

Przy użyciu linii adresowych do A19 włącznie, tryb ten daje dostęp do obszarów:

```
$00000 --    $07FFF   i      $F8000 --    $FFFFF
```

Mimo to, wewnętrzna reprezentacja adresów jest nadal 32-bitowa:

```
$00000000 -- $00007FFF   i   $FFFF8000 -- $FFFFFFFF
```

Tryb natychmiastowy (*immediate*)

Kodowanie:

pole trybu: %111, pole rejestru: %100

```
STALA       EQU     $55AA

            MOVE.W  #STALA,D2                   STALA -> D2
```

Rozmiar danych (domyślny: `.W`):

- `.W` – słowo 16-bitowe, $EA = (PC) + 2$
- `.L` – długie słowo (32-bitowe), $EA = (PC) + 2$
- `.B` – 8-bitowy bajt, $EA = (PC) + 3$

Krótka odmiana trybu (w niektórych rozkazach):

- `MOVEQ` – 8-bitowego pole danych w słowie rozkazowym,
- `ADDQ, SUBQ` – uogólnienie inkrementacji i dekrementacji – 3-bitowe pola.

Tryb pośredni rejestru adresowego (*address register indirect*)

$$EA = (An)$$

Przykład (tryb bezpośredni – 4 słowa, 5 cykli):

```
Dane    EQU     $55AA           #define Dane 0x55aa
PortAdd EQU     $FFF000         #define PortAdd 0xfff000
* Deklaracja zmiennych
        ORG     PortAdd
Port    DS.W    1               short Port @ Portadd;
* Przeslanie danych do portu
        MOVE.W  #Dane,Port      Port = Dane;
```

Tryb pośredni – 2 słowa, 3 cykle:

```
Dane    EQU     $55AA           #define Dane 0x55aa
PortAdd EQU     $FFF000         #define PortAdd 0xfff000
* Inicjalizacja wskaznika      short *ptr; /* rejestr A0 */
        ORG     ROM
        MOVEA.L #PortAdd,A0     ptr = (short *)PortAdd;
* Przeslanie danych do portu
        MOVE.W  #Dane,(A0)      *ptr = Dane;
```

Tryb pośredni z autopostinkrementacją (*address register indirect with postincrement*)

$$EA = (An)$$
$$(An) + SIZE \rightarrow An$$

Kodowanie:

pole trybu: %011, wskaźnik: rejestr adresowy, $SIZE$: ".B" – 1, ".W" – 2, ".L" – 4.

Przykład:

```
* Deklaracje zmiennych
        ORG     RAM
*                               short *ptrA; /* rejestr A0 */
*                               short *ptrB; /* rejestr A1 */
TablA   DS.W    100             short TablA[100];
TablB   DS.W    100             short TablB[100];
* Inicjalizacja wskaznikow
        ORG     ROM
        MOVEA.L #TablA,A0       ptrA = &TablA[0];
        MOVEA.L #TablB,A1       ptrB = &TablB[0];
* Przepisanie elementow i przesuniecie wskaznikow
        MOVE.W  (A0)+,(A1)+     *ptrB++ = *ptrA++;
```

Tryb pośredni z autopredekrementacją (*address register indirect with predecrement*)

$$(An) - SIZE \rightarrow An$$
$$EA = (An)$$

Kodowanie:

pole trybu: %100, wskaźnik: rejestr adresowy, $SIZE$: " .B" – 1, " .W" – 2, " .L" – 4.

Przykład zastosowania:

przepisywanie tablic w odwrotnej kolejności (przy częściowym pokrywaniu się obszaru źródłowego z docelowym):

```
MOVE.W  -(A0),-(A1)      *--ptrB = *--ptrA;
```

Uwaga:

Ważne zastosowanie – operacje na stosie: instrukcja `MOVEM` (*MOVE Multiple*) pozwala przesłać dowolny podzbiór rejestrów danych i adresowych procesora przy pomocy rozkazu złożonego z dwóch słów: kodu rozkazu i maski zestawu rejestrów.

Tryb pośredni z przesunięciem (*address register indirect with displacement*)

$$EA = (An) + ((PC) + 2)_{SEX}$$

Przykład:

```
* Definicja typu data_t
        ORG     0               typedef struct{
Dzien   DS.B    1                 char Dzien;
Miesiac DS.B    1                 char Miesiac;
Rok     DS.W    1                 short Rok;
*                               } data_t;
Data_L  EQU     *               Data_L = sizeof(data_t);
* Deklaracje zmiennych
        ORG     RAM             data_t *ptr; /* w A0 */
Data    DS.B    Data_L          data_t Data;
* Dostep do struktury
        ORG     ROM
        MOVEA.L #Data,A0        ptr = &Data;
        MOVE.W  #1996,Rok(A0)   ptr->Rok = 1996;
```

Tryb pośredni z indeksem i przesunięciem (*address register indirect with index and displacement*)

$$EA = (An) + ((PC) + 3)_{SEX} + (Xm)_{SEX}$$

Przykład 1 – bez skalowania (tablica $char$):

```
* Definicja typu pomiar_t
         ORG     0                  typedef struct{
Datas    DS.B    Data_L               data_t Datas;
Temp     DS.B    24                   char Temp[24];
*                                   } pomiar_t;
Pomiar_L EQU     *                  Pomiar_L=sizeof(pomiar_t);
* Deklaracje zmiennych
         ORG     RAM
*                                   pomiar_t *ptr; /* w A0 */
*                                   int NumerPom;  /* w D1 */
*                                   char Wynik;    /* w D0 */
Pomiar   DS.B    Pomiar_L           pomiar_t Pomiar;
* Dostep do elementu tablicy wewnatrz struktury
         ORG     ROM
         MOVEA.L #Pomiar,A0         ptr = &Pomiar;
         MOVE.L  #0,D1              NumerPom = 0;
* odczyt pomiaru z tablicy
         MOVE.B  Temp(A0,D1.L),D0   Wynik=ptr->Temp[NumerPom];
```

$$EA = (An) + ((PC) + 3)_{SEX} + (Xm)_{SEX} \star SCALE$$

Przykład 2 – ze skalowaniem (tablica $long$):

```
* Definicja typu pomiar_t
         ORG     0                    typedef struct{
Datas    DS.B    Data_L                 data_t Datas;
Temp     DS.L    24                     long Temp[24];
*                                       } pomiar_t;
Pomiar_L EQU     *                      Pomiar_L=sizeof(pomiar_t);
* Deklaracje zmiennych
         ORG     RAM
*                                       pomiar_t *ptr; /* w A0 */
*                                       int NumerPom;  /* w D1 */
*                                       char Wynik;    /* w D0 */
Pomiar   DS.B    Pomiar_L               pomiar_t Pomiar;
* Dostep do elementu tablicy wewnatrz struktury
         ORG     ROM
         MOVEA.L #Pomiar,A0             ptr = &Pomiar;
         MOVE.L  #0,D1                  NumerPom = 0;
* odczyt pomiaru z tablicy
         MOVE.L  Temp(A0,D1.L*4),D0 Wynik=ptr->Temp[NumerPom];
```

Kodowanie indeksu

Wersja 0 (bit 8 = 0):

pole trybu: %110, pole rejestru: numer rejestru bazowego, kolejne słowo: parametry rejestru indeksowego, skala i 8-bitowe przesunięcie bazowe:

```
bit:  15 14 13 12 11 10  9   8   7  6  5  4  3  2  1  0
       0  0  0  1  1  1  0   0   0  0  0  0  0  1  0  0
      D/A <-----> W/L <-->       <--------------------->
      typ/numer/czesc skala       przesuniecie bazowe
      rej. indeks.    elem.
```

Wersja 1 (bit 8 = 1):

pole trybu: %110, pole rejestru: numer rejestru bazowego, kolejne słowo: parametry rejestru indeksowego, skala i opcje podtrybów (ew. przesunięcie bazowe w nastepnych słowach):

```
bit:  15 14 13 12 11 10  9   8   7  6  5  4  3  2  1  0
       0  0  0  1  1  1  0   1   0  0  0  0  0  1  0  0
      D/A <-----> W/L <-->      BS IS  <-->     <----->
      typ/numer/czesc skala      |  | rozmiar     I/IS
      rej. indeks.    elem.      |  | przes.
                                 |  |__ brak indeksowania
                                 |_____ brak rej. bazowego
```

Pole	Definicja
BS	wykluczenie rejestru bazowego *Base register Suppress* 0 = jest rejestr bazowy 1 = brak rejestru bazowego
IS	wykluczenie indeksu *Index Suppress* 0 = jest indeksowanie 1 = brak indeksowania
BD SIZE	rozmiar przesunięcia bazowego *Base Displacement SIZE* 00 = zarezerwowane 01 = zerowe przesunięcie 10 = 16-bitowe przesunięcie 11 = 32-bitowe przesunięcie
I/IS	wybór podtrybów *Index/Indirect Selection* wraz z polem IS opisane w następnej tablicy

IS	I/IS	Opis trybu
0	000	brak odwołania pośredniego
0	001	brak w CPU32 (preindeksacja bez zewnętrznego przesunięcia)
0	010	brak w CPU32 (preindeksacja z 16-bitowym zewnętrznym przesunięciem)
0	011	brak w CPU32 (preindeksacja z 32-bitowym zewnętrznym przesunięciem)
0	100	zarezerwowane
0	101	brak w CPU32 (postindeksacja bez zewnętrznego przesunięcia)
0	110	brak w CPU32 (postindeksacja z 16-bitowym zewnętrznym przesunięciem)
0	111	brak w CPU32 (postindeksacja z 32-bitowym zewnętrznym przesunięciem)
1	000	brak odwołania pośredniego
1	001	pośrednie bez zewnętrznego przesunięcia
1	010	pośrednie z 16-bitowym zewnętrznym przesunięciem
1	011	pośrednie z 32-bitowym zewnętrznym przesunięciem
1	100–111	zarezerwowane

Tryb pośredni PC z przesunięciem (*PC indirect with displacement*)

$$EA = (PC) + 2 + ((PC) + 2)_{SEX}$$

Odwołanie do danych zawartych w programie; kod niezależny od położenia (*PIC – Position Independent Code*):

```
        ORG     ROM
Napis   DC.B    'Ala ma kota'   stala tablica znakowa
        DC.B    0
*
* Dostep do pierwszego znaku napisu
        MOVE.B  Napis(PC),D0
```

Przykład z instrukcją `LEA` (*Load Effective Address*):

```
        ORG     ROM
Napis   DC.B    'Ala ma kota'   stala tablica znakowa
        DC.B    0
*
* Inicjalizacja rejestru bazowego
        LEA.L   Napis(PC),A0    (PC)+offset->A0
*
* Dostep do kolejnych znakow napisu
        MOVE.B  (A0)+,Port
```

Tryb pośredni PC z indeksem i przesunięciem (*PC indirect with index and displacement*)

$$EA = (PC) + 2 + ((PC) + 3)_{SEX} + (Xm)_{SEX}$$

Przykład (tablica konwersji umieszczona w kodzie programu); kod niezależny od położenia (PIC – $Position\ Independent\ Code$**):**

```
        ORG     ROM
* Deklaracja danych stalych          const char
TabPi   DC.B    1,2,3,5               TabPi[6] = {1,2,3,5};

* Inicjalizacja indeksu w tablicy
        MOVE.W  #5,D1                 NrLiczby = 5;

* Odczyt liczby z tablicy
        MOVE.B  TabPi(PC,D1.W),D0     Wynik=TabPi[NrLiczby];
```

Rozszerzenia trybów adresowania w M68K, niedostępne w CPU32

Pośrednie tryby adresowania z indeksowaniem:

- *memory indirect postindexed*

$$EA = ((An) + bd_{SEX}) + (Xm)_{SEX} \star SCALE + od_{SEX}$$

- *memory indirect preindexed*

$$EA = ((An) + bd_{SEX} + (Xm)_{SEX} \star SCALE) + od_{SEX}$$

- *program counter memory indirect postindexed*

$$EA = ((PC) + 2 + bd_{SEX}) + (Xm)_{SEX} \star SCALE + od_{SEX}$$

- *program counter memory indirect preindexed*

$$EA = ((PC) + 2 + bd_{SEX} + (Xm)_{SEX} \star SCALE) + od_{SEX}$$

Przykładowy listing asemblacji

```
00000000 =00F00000         1  ROM     EQU     $f00000
00000000 =00001000         2  RAM     EQU     $1000
00000000                   3
00000000                   4  * Tryb posredni rejestru adresowego
00000000                   5
00000000 =000055AA         6  Dane    EQU     $55AA
00000000 =00FFF000         7  PortAdd EQU     $FFF000
00000000                   8
00000000                   9  * Deklaracje zmiennych
00000000                  10  *
00000000                  11
00000000                  12  * Inicjalizacja wskaznika
00F00000                  13          ORG     ROM
00F00000 207C 00FFF000    14          MOVEA.L #PortAdd,A0
00F00006                  15
00F00006                  16  * Przeslanie danych do portu
00F00006 30BC 55AA        17          MOVE.W  #Dane,(A0)
00F0000A                  18
00F0000A                  19
00F0000A                  20  * Tryb posredni rejestru adresowego
00F0000A                  21  * (address register indirect with po
00F0000A                  22
```

```
00F0000A                    23  * Deklaracje zmiennych
00001000                    24          ORG     RAM
00001000                    25  *
00001000                    26  *
00001000                    27  TablA   DS.W    100
000010C8                    28  TablB   DS.W    100
00001190                    29
00001190                    30  * Inicjalizacja wskaznikow
00F00000                    31          ORG     ROM
00F00000 207C 00001000      32          MOVEA.L #TablA,A0
00F00006 227C 000010C8      33          MOVEA.L #TablB,A1
00F0000C                    34
00F0000C                    35  * Przepisanie elementow i przesuniec
00F0000C 32D8               36          MOVE.W  (A0)+,(A1)+
00F0000E                    37
00F0000E                    38
00F0000E                    39  * Tryb posredni rejestru adresowego
00F0000E                    40  * (address register indirect with pr
00F0000E                    41
00F0000E                    42  * Przesuniecie wskaznikow i przepisa
00F0000E 3320               43          MOVE.W  -(A0),-(A1)
00F00010                    44
00F00010                    45
00F00010                    46  * Tryb posredni rejestru adresowego
00F00010                    47  * (address register indirect with di
00F00010                    48
```

```
00F00010                    49  * Definicja struktury data_t
00000000                    50          ORG     0
00000000                    51  *
00000000                    52  Dzien   DS.B    1
00000001                    53  Miesiac DS.B    1
00000002                    54  Rok     DS.W    1
00000004                    55  *
00000004                    56  Data_L
00000004                    57  *
00000004                    58
00000004                    59  * Deklaracje zmiennych
00001000                    60          ORG     RAM
00001000                    61  *
00001000                    62  Data    DS.B    Data_L
00001004                    63
00001004                    64  * Dostep do wnetrza struktury
00F00000                    65          ORG     ROM
00F00000 207C 00001000      66          MOVEA.L #Data,A0
00F00006 317C 07CC 0002     67          MOVE.W  #1996,Rok(A0)
00F0000C                    68
00F0000C                    69
00F0000C                    70  * Tryb posredni rejestru adresowego
00F0000C                    71  * (address register indirect with in
00F0000C                    72
00F0000C                    73  * Definicja struktury pomiar_t
00000000                    74          ORG     0
```

```
00000000                    75  *
00000000                    76  Datas   DS.B    Data_L
00000004                    77  Temp    DS.B    24
0000001C                    78  *
0000001C                    79  Pomiar_L
0000001C                    80  *
0000001C                    81
0000001C                    82  * Deklaracje zmiennych
00001000                    83          ORG     RAM
00001000                    84  *
00001000                    85  *
00001000                    86  *
00001000                    87  Pomiar  DS.B    Pomiar_L
0000101C                    88
0000101C                    89  * Dostep do elementu tablicy wewnatr
00F00000                    90          ORG     ROM
00F00000                    91
00F00000                    92  * Inicjalizacja wskaznika struktury
00F00000 207C 00001000      93          MOVEA.L #Pomiar,A0
00F00006                    94
00F00006                    95  * Inicjalizacja indeksu w tablicy
00F00006 7200               96          MOVE.L  #0,D1
00F00008                    97
00F00008                    98  * Odczyt pomiaru z tablicy
00F00008 1030 1804          99          MOVE.B  Temp(A0,D1.L),D0
00F0000C                   100
```