```
;*********************************************************************
;****************** Date: July 22, 2009 - Writing ********************
;****************** Test  Programmable [Matrix  5x7] *****************
;****************               Animation               ***************
;*************** Designed By: Esmail Bakhshzad Mahmodi ****************
;*********************************************************************
; Processor: ATmega8 ( 8-bit with,8K Bytes In-System,Programmable Flash  *
; Integrated Circuit1: 74LS573 (Octal D Latch with 3-STATE Outputs)      *
; Integrated Circuit2: ULN2803 (HIGH-VOLTAGE,HIGH-CURRENT,8-bit Array    *
; Compiler: AVR Studio 4                                                 *
; E-mail: Esmail_bakhshzad@yahoo.com                                     *
;*********************************************************************
; 1- PC0 - PC4 Connected to IC(ULN2803) Column Output For Data Pins
; 2- PB0 Connected to LE=> Latch Enable Input (Active HIGH)
; 3- PD0 - PD7 Connected to IC(74LS573) Line Output For Data Pins
;Clock frequency processor in cycles/s 8MHz = 8,000,000 Cycle / S
;Programmable Subroutine Matrix 5x7
; ====================================================================
;                          Hardware Connections
; ====================================================================
;              ___   ___
;     RESET |1  |_| 28| PC5
; Row7  PD0 |2   A  27| PC4 Column5
; Row6  PD1 |3   T  26| PC3 Column4
; Row5  PD2 |4   M  25| PC2 Column3
; Row4  PD3 |5   E  24| PC1 Column2
; Row3  PD4 |6   L  23| PC0  Column1
;       VCC |7      22| GND
;       GND |8   A  21| AREF (+2.56 V, output)
;     XTAL1 |9   T  20| AVCC input
;     XTAL2 |10  m  19| PB5
; Row2  PD5 |11  e  18| PB4
; Row1  PD6 |12  g  17| PB3 Latch4
;       PD7 |13  a  16| PB2 Latch3
;Latch1 PB0 |14  8  15| PB1 Latch2
;           |_________|
;
;--------------------------------------------------------------------
.NOLIST                         ;Disable listfile generation(Combination)
.include "m8def.inc"            ;Read source from another file
.LIST                           ;Reenable listfile generation on
;--------------------------------------------------------------------
;-*************** ( Define Ports Local Pointer Variable ) ***************-
;--------------------------------------------------------------------
.equ    Count =0xFC             ;Set a symbol equal to an expression
.equ    Storage =0x0060         ;
.def    Status  =R0             ;define a storage for Status=R0
.def    Var =R16                ;define a storage register R16=variable
.def    Column =R17             ;define a storage for Column=R17
.def    Scan =R18               ;define a storage for Scan=R18
.def    Counter =R19            ;define a storage for Counter=R19
.def    Flags =R20              ;define a storage for Flags=R20[Com keys]
.def    Temp1 =R21              ;define a storage for Temp1=R21
.def    Temp2 =R22              ;define a storage for Temp1=R22
.def    Temp3 =R23              ;define a storage for Temp1=R23
.def    Temp4 =R24              ;define a storage for Temp1=R24
.def    Refresh =R25            ;define a storage for Refresh=R25
.def    Character =R26          ;define a storage for Character=R26
.def    Animation =R27          ;define a storage for Animation=R27
;define ports in/out-------------------------------------------------
.equ    ColOut =PORTC           ;Output and Pull-Up-PortC Output= Column5
.equ    ColDdr =DDRC            ;Data direction register of the portC
;--------------------------------------------------------------------
.equ    RowOut =PORTD           ;Output and Pull-Up-PortD *** output= Row7
.equ    RowDdr =DDRD            ;Data direction register of the portD
;--------------------------------------------------------------------
;Latch-PortB Output
.equ    Latch1 =0               ;bit in I/O register PORTB.0
.equ    Latch2 =1               ;bit in I/O register PORTB.1
.equ    Latch3 =2               ;bit in I/O register PORTB.2
.equ    Latch4 =3               ;bit in I/O register PORTB.3
```

```
;RELAY key PortB Input
.equ    RELkey1 =4              ;bit in I/O register PORTB.4
.equ    RELkey2 =5              ;bit in I/O register PORTB.5
.equ    RELkey3 =6              ;bit in I/O register PORTB.6
;---------------------------**** START ****------------------------------
        .CSEG
START:  .ORG    $0000
;---------------------------**** MACRO ****------------------------------
.LISTMAC
.MACRO Stack Pointer            ;Define an macro [Stack Pointer]
        LDI     Counter,0x00    ;Load Immediate register R19-00 hex
        OUT     SPL,Counter     ;Stack Pointer Low 00 hex
        LDI     Counter,0x01    ;Load Immediate register R19-01 hex
        OUT     SPH,Counter     ;Stack Pointer High 01 hex
.ENDMACRO                       ;End macro definition
.MACRO Delay_25ms               ;Define an macro [Delay_25ms]
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   Provide         ;Call Subroutine overflow (25ms) Provide
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   Provide         ;Call Subroutine overflow (25ms) Provide
.ENDMACRO
;-------------------------**** ENDMACRO ****----------------------------
;-----------------------------------------------------------------------
;-***************** (Program FUSE BITS Specify Device) ******************-
;-----------------------------------------------------------------------
;-----------------------------------------------------------------------
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   Reset           ;Call Subroutine Reset handler    C->3
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   Provide         ;Call Subroutine overflow (25ms) Provide
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   Main            ;Call Subroutine Main
;-----------------------------------------------------------------------
;-**************** (Program Subroutine Reset handler) ******************-
;-----------------------------------------------------------------------
;To Provide Initial Port,Interrupt Setting Up
Reset:
        CLR     Var             ;Load Immediate register R16=00 hex
        OUT     ColOut,Var      ;Make PORTC For Output
        OUT     ColDdr,Var      ;Make DDRC For Output
        OUT     RowOut,Var      ;Make PORTD For Output
        OUT     RowDdr,Var      ;Make DDRD For Output
        SBI     PORTB,Latch1    ;Set bit in I/O register PORTB.0
        SBI     PORTB,Latch2    ;Set bit in I/O register PORTB.1
        SBI     PORTB,Latch3    ;Set bit in I/O register PORTB.2
        SBI     PORTB,Latch4    ;Set bit in I/O register PORTB.3
        CBI     PORTB,RELkey1   ;Clear bit in I/O register PORTB.4
        CBI     PORTB,RELkey2   ;Clear bit in I/O register PORTB.5
        CBI     PORTB,RELkey3   ;Clear bit in I/O register PORTB.6
        CLR     Status          ;Load Immediate register R0=00 hex
        CLR     Column          ;Load Immediate register R17=00 hex
        CLR     Scan            ;Load Immediate register R18=00 hex
        CLR     Counter         ;Load Immediate register R19=00 hex
        CLR     Flags           ;Load Immediate register R20=00 hex
        CLR     Temp1           ;Load Immediate register R21=00 hex
        CLR     Temp2           ;Load Immediate register R22=00 hex
        CLR     Temp3           ;Load Immediate register R23=00 hex
        CLR     Temp4           ;Load Immediate register R24=00 hex
        CLR     Refresh         ;Load Immediate register R25=00 hex
        CLR     Character       ;Load Immediate register R26=00 hex
        CLR     Animation       ;Load Immediate register R27=00 hex
        RET
;-----------------------------------------------------------------------
;-****************** (Program Subroutine Animation) *********************-
;-----------------------------------------------------------------------
Main:   CLR     Column          ;Load Immediate register R17=00 hex
        CLR     Scan            ;Load Immediate register R18=00 hex
        CLR     Counter         ;Load Immediate register R19=00 hex
        CLR     Temp1           ;Load Immediate register R21=00 hex
        CLR     Temp2           ;Load Immediate register R22=00 hex
        CLR     Temp3           ;Load Immediate register R23=00 hex
```

```
        CLR     Temp4           ;Load Immediate register R24=00 hex
        CLR     Refresh         ;Load Immediate register R25=00 hex
        CLR     Character       ;Load Immediate register R26=00 hex
        CLR     Animation       ;Load Immediate register R27=00 hex
        LDI     Var,0xFF        ;Load Immediate register R16= FF hex
        OUT     ColDdr,Var      ;Make DDRC For Output= FF hex
        OUT     RowDdr,Var      ;Make DDRD For Output= FF hex
;Device= Enable Watchdog Times
        LDI     Temp1,0x1E      ;Load Immediate register R21=0D hex
        OUT     WDTCR,Temp1     ;Enable Watchdog Time out 0.55 s
;** part of byte address ** -------------------------------------------
;*************
Charac: Delay_25ms
        Delay_25ms
        INC     Character       ;Increment Register R26=00+1
        CPI     Character,0x5   ;Compare Register with Immediate Rd=5
        BREQ    Main            ;Branch if Equal Jump Main
        CPI     Character,0x1   ;Compare Register with Immediate Rd=1
        BREQ    Character1      ;Branch if Equal Jump Character1
        CPI     Character,0x2   ;Compare Register with Immediate Rd=2
        BREQ    Character2      ;Branch if Equal Jump Character2
        CPI     Character,0x3   ;Compare Register with Immediate Rd=3
        BREQ    Character3      ;Branch if Equal Jump Character3
        CPI     Character,0x4   ;Compare Register with Immediate Rd=4
        BREQ    Character4      ;Branch if Equal Jump Character4
;*****************************************************************
Character1:
        LDI     ZH,high(2*Symbol3);Load high part of byte address into R31
        LDI     ZL,low(2*Symbol3);Load low part of byte address into R30
        RJMP    Display         ;Relative Jump to Display
Character2:
        LDI     ZH,high(2*Symbol1);Load high part of byte address into R31
        LDI     ZL,low(2*Symbol1);Load low part of byte address into R30
        RJMP    Display         ;Relative Jump to Display
Character3:
        LDI     ZH,high(2*Symbol4);Load high part of byte address into R31
        LDI     ZL,low(2*Symbol4);Load low part of byte address into R30
        RJMP    Display         ;Relative Jump to Display
Character4:
        LDI     ZH,high(2*Symbol2);Load high part of byte address into R31
        LDI     ZL,low(2*Symbol2);Load low part of byte address into R30
        RJMP    Display         ;Relative Jump to Display
;*****************************************************************
Display:CLR     Column          ;Load Immediate register R17=00 hex
        LDI     Scan,0b00000001 ;Load Immediate register R18=01 hex
Loop_2: LPM                     ;Load byte from program memory into R0
Count_1:INC     Column          ;Increment Register R17=00+1
        CPI     Column,0x06     ;Compare Register with Immediate Rd=06
        BREQ    Exit_1          ;Branch if Equal Jump Exit_1
        OUT     ColOut,Scan     ;Make PORTC For Output=Scan
        LSL     Scan            ;Logical Shift Left 00000010 <-- 00000001
        TST     R0              ;Check if reached the end of the Write=R0
        OUT     RowOut,R0       ;Make PORTD For Output=R0
        BREQ    Quit            ;If so , Quit (Test for Zero or Minus)
        ADIW    ZL,1            ;Increase Z registers
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   Delay_20us      ;Call Subroutine Overflow Delay_20us  C->3
        LDI     Var,0x00        ;Load Immediate register R16= 00 hex
        OUT     ColOut,Var      ;Make PORTC For Output=Temp4
        OUT     RowOut,Var      ;Make PORTD For Output=Temp4
        Stack Pointer           ;Call macro, show [Stack Pointer]
        RCALL   WDT_off         ;Call Subroutine Overflow  WDT_off
        RJMP    Loop_2          ;Relative Jump to Loop_2
Quit:   RJMP    Count_1         ;Relative Jump to Count_1
Exit_1: RJMP    Charac          ;Relative Jump to Charac
;--------------------**** Watchdog Times OFF ****-------------------------
WDT_off:
        WDR                     ;reset WDT
        IN       R16,WDTCR      ;Write logical one to WDCE and WDE
        ORI      R16,(1<<WDCE)|(1<<WDE)
        OUT      WDTCR,R16      ;Load Immediate register
```

```
        ldi      R16, (0<<WDE)  ;Turn off WDT
        OUT      WDTCR, R16     ;Load Immediate register
        RET
;---------------------------------------------------------------------------
;-************ (Timer Overflow Interrupt service routine) ***************-
;---------------------------------------------------------------------------
;Updates 25 ms, flash and debounce counter to provide Time reference
;0- Temp1=256
;1- Temp2=208
;2- cycles/s 8 MHz => 1/8 MHz = 0.125 us
;3- Computation cycle >> Loop_5 >> (1+0.5)208*0.125us = 39 us
;4- Computation cycle >> Provide >> (1+1+0.5)256*39us = 24.96 ms
;5- RCALL + RET = 3+4 * 0.125 us = 0.875 us -- >>> 24.960875 ms
;C --> Engine Cycles
Provide:
        LDI     Temp2,208       ;Load Immediate register R22=D0 hex C->1
Loop_5: DEC     Temp2           ;Decrement Register Temp2 = FF-1    C->1
        BRNE    Loop_5          ;Branch if not Equal Loop_5         C->0.5
        DEC     Temp1           ;Load Immediate register R21=256    C->1
        BRNE    Provide         ;Branch if not Equal Provide        C->0.5
        RET                     ;Subroutine Return                  C->4

;---------------------------------------------------------------------------
;-********************** (Give Some Time 20 us) **********************-
;---------------------------------------------------------------------------
;Time reference Give Some Time 20us
;0- TCCR0 = 0000 0010  ClkI/O/8 (From prescaler) 8MHz / 8 = 1MHz
;2- T = 1/1MHz = 1 us
;3- Time 20us = 20us/1us =20
;4- TCNT0 = ? >>>  256-20=236 >>> hex >>> FC
Delay_20us:
        LDI     Temp1,0x02      ;Load Immediate register R21=02 hex C->1
        OUT     TCCR0,Temp1     ;Timer/Counter0 Control Registe     C->1
        LDI     Temp2,Count     ;Load Immediate register R22=FC hex C->1
        OUT     TCNT0,Temp2     ;Timer/Counter0 (8 Bits) =>FC hex   C->1
;TCCR0=010     ClkI/O/8 (From prescaler)
Loop_20us:
        IN      Temp3,TIFR      ;Timer/Counter Interrupt FlagRegister
        SBRS    Temp3,TOV0      ;Skip if Bit in Register is Set TOV0=1
        RJMP    Loop_20us       ;Jump to Main
        LDI     Temp3,0xFF      ;Load Immediate register R23=FF hex C->1
        OUT     TIFR,Temp3      ;Clear Interrupt FlagRegister       C->1
        RET                     ;Subroutine Return                  C->4
       ;END                     ;End of File Marker
;$$$$$$$$$$$$$$$$$$$$ ( Write Address Crossword ) $$$$$$$$$$$$$$$$$$$$
;Subroutine Write Data Crossword
; | / - \
Symbol3:
        .DB     $80,$80,$7F,$00,$00,$00,$00,$00     ;Code  Symbol3 '|'
Symbol1:
        .DB     $02,$04,$08,$10,$20,$00,$00,$00     ;Code  Symbol1 '/'
Symbol4:
        .DB     $08,$08,$08,$08,$08,$00,$00,$00     ;Code  Symbol4 '-'
Symbol2:
        .DB     $20,$10,$08,$04,$02,$00,$00,$00     ;Code  Symbol2 '\'

;Write:
;       .DB     $3F,$44,$44,$44,$3F,$00,$00,$00     ;Code  Word1  'A'
;       .DB     $02,$15,$15,$15,$0F,$00,$00,$00     ;Code  SWor1  'a'
;       .DB     $7F,$49,$49,$49,$36,$00,$00,$00     ;Code  Word2  'B'
;       .DB     $7F,$09,$11,$11,$0E,$00,$00,$00     ;Code  SWor2  'b'
;       .DB     $3E,$41,$41,$41,$22,$00,$00,$00     ;Code  Word3  'C'
;       .DB     $0E,$11,$11,$11,$02,$00,$00,$00     ;Code  SWor3  'c'
;       .DB     $7F,$41,$41,$22,$1C,$00,$00,$00     ;Code  Word4  'D'
;       .DB     $0E,$11,$11,$09,$7F,$00,$00,$00     ;Code  SWor4  'd'
;       .DB     $7F,$49,$49,$49,$41,$00,$00,$00     ;Code  Word5  'E'
;       .DB     $0E,$15,$15,$15,$0C,$00,$00,$00     ;Code  SWor5  'e'
;       .DB     $7F,$48,$48,$48,$40,$00,$00,$00     ;Code  Word6  'F'
;       .DB     $08,$3F,$48,$40,$20,$00,$00,$00     ;Code  SWor6  'f'
;       .DB     $3E,$41,$41,$49,$2F,$00,$00,$00     ;Code  Word7  'G'
;       .DB     $18,$25,$25,$25,$3E,$00,$00,$00     ;Code  SWor7  'g'
```

```
;       .DB     $7F,$08,$08,$08,$7F,$00,$00,$00     ;Code  Word8  'H'
;       .DB     $7F,$08,$10,$10,$0F,$00,$00,$00     ;Code  SWor8  'h'
;       .DB     $80,$41,$7F,$41,$00,$00,$00,$00     ;Code  Word9  'I'
;       .DB     $80,$11,$5F,$01,$00,$00,$00,$00     ;Code  SWor9  'i'
;       .DB     $02,$01,$41,$7E,$40,$00,$00,$00     ;Code  Word10 'J'
;       .DB     $02,$01,$11,$5E,$00,$00,$00,$00     ;Code  SWor10 'j'
;       .DB     $7F,$08,$14,$22,$41,$00,$00,$00     ;Code  Word11 'K'
;       .DB     $7F,$04,$0A,$11,$00,$00,$00,$00     ;Code  SWor12 'k'
;       .DB     $7F,$01,$01,$01,$01,$00,$00,$00     ;Code  Word12 'L'
;       .DB     $80,$41,$7F,$01,$00,$00,$00,$00     ;Code  SWor12 'l'
;       .DB     $7F,$20,$18,$20,$7F,$00,$00,$00     ;Code  Word13 'M'
;       .DB     $1F,$10,$0C,$10,$0F,$00,$00,$00     ;Code  SWor13 'm'
;       .DB     $7F,$10,$08,$04,$7F,$00,$00,$00     ;Code  Word14 'N'
;       .DB     $1F,$08,$10,$10,$0F,$00,$00,$00     ;Code  SWor14 'n'
;       .DB     $3E,$41,$41,$41,$3E,$00,$00,$00     ;Code  Word15 'O'
;       .DB     $0E,$11,$11,$11,$0E,$00,$00,$00     ;Code  SWor15 'o'
;       .DB     $7F,$48,$48,$48,$30,$00,$00,$00     ;Code  Word16 'P'
;       .DB     $3F,$28,$28,$28,$10,$00,$00,$00     ;Code  SWor16 'p'
;       .DB     $3E,$41,$45,$42,$3D,$00,$00,$00     ;Code  Word17 'Q'
;       .DB     $10,$28,$28,$18,$3F,$00,$00,$00     ;Code  SWor17 'q'
;       .DB     $7F,$48,$4C,$4A,$31,$00,$00,$00     ;Code  Word18 'R'
;       .DB     $1F,$08,$10,$10,$08,$00,$00,$00     ;Code  SWor18 'r'
;       .DB     $31,$49,$49,$49,$46,$00,$00,$00     ;Code  Word19 'S'
;       .DB     $09,$15,$15,$15,$02,$00,$00,$00     ;Code  SWor19 's'
;       .DB     $40,$40,$7F,$40,$40,$00,$00,$00     ;Code  Word20 'T'
;       .DB     $10,$7E,$11,$01,$02,$00,$00,$00     ;Code  SWor20 't'
;       .DB     $7E,$01,$01,$01,$7E,$00,$00,$00     ;Code  Word21 'U'
;       .DB     $1E,$01,$01,$02,$1F,$00,$00,$00     ;Code  SWor21 'u'
;       .DB     $7C,$02,$01,$02,$7C,$00,$00,$00     ;Code  Word22 'V'
;       .DB     $1C,$02,$01,$02,$1C,$00,$00,$00     ;Code  SWor22 'v'
;       .DB     $7E,$01,$06,$01,$7E,$00,$00,$00     ;Code  Word23 'W'
;       .DB     $1E,$01,$06,$01,$1E,$00,$00,$00     ;Code  SWor23 'w'
;       .DB     $63,$14,$08,$14,$63,$00,$00,$00     ;Code  Word24 'X'
;       .DB     $11,$0A,$04,$0A,$11,$00,$00,$00     ;Code  SWor24 'x'
;       .DB     $70,$08,$07,$08,$70,$00,$00,$00     ;Code  Word25 'Y'
;       .DB     $18,$05,$05,$05,$1E,$00,$00,$00     ;Code  SWor25 'y'
;       .DB     $43,$45,$49,$51,$61,$00,$00,$00     ;Code  Word26 'Z'
;       .DB     $11,$13,$15,$19,$11,$00,$00,$00     ;Code  SWor26 'z'
;       .DB     $3E,$45,$49,$51,$3E,$00,$00,$00     ;Code  Number0 '0'
;       .DB     $80,$21,$7F,$01,$00,$00,$00,$00     ;Code  Number1 '1'
;       .DB     $21,$43,$45,$49,$31,$00,$00,$00     ;Code  Number2 '2'
;       .DB     $42,$41,$51,$69,$46,$00,$00,$00     ;Code  Number3 '3'
;       .DB     $0C,$14,$24,$7F,$04,$00,$00,$00     ;Code  Number4 '4'
;       .DB     $72,$51,$51,$51,$4E,$00,$00,$00     ;Code  Number5 '5'
;       .DB     $1E,$29,$49,$49,$06,$00,$00,$00     ;Code  Number6 '6'
;       .DB     $40,$47,$48,$50,$60,$00,$00,$00     ;Code  Number7 '7'
;       .DB     $36,$49,$49,$49,$36,$00,$00,$00     ;Code  Number8 '8'
;       .DB     $30,$49,$49,$4A,$3C,$00,$00,$00     ;Code  Number9 '9'
;       .DB     $62,$64,$08,$13,$23,$00,$00,$00     ;Code  Symbol0 '%'
;       .DB     $02,$04,$08,$10,$20,$00,$00,$00     ;Code  Symbol1 '/'
;       .DB     $20,$10,$08,$04,$02,$00,$00,$00     ;Code  Symbol2 '\'
;       .DB     $80,$80,$7F,$00,$00,$00,$00,$00     ;Code  Symbol3 '|'
;       .DB     $08,$08,$08,$08,$08,$00,$00,$00     ;Code  Symbol4 '-'
;       .DB     $08,$08,$3E,$08,$08,$00,$00,$00     ;Code  Symbo15 '+'
;       .DB     $14,$14,$14,$14,$14,$00,$00,$00     ;Code  Symbo16 '='
;       .DB     $14,$08,$3E,$08,$14,$00,$00,$00     ;Code  Symbo17 '*'
;       .DB     $01,$01,$01,$01,$01,$00,$00,$00     ;Code  Symbol8 '_'
;       .DB     $80,$1C,$22,$41,$00,$00,$00,$00     ;Code  Symbol9 '('
;       .DB     $80,$41,$22,$1C,$00,$00,$00,$00     ;Code  Symbo20 ')'
;       .DB     $80,$7F,$41,$41,$00,$00,$00,$00     ;Code  Symbo21 '['
;       .DB     $80,$41,$41,$7F,$00,$00,$00,$00     ;Code  Symbo22 ']'
;       .DB     $80,$41,$22,$14,$08,$00,$00,$00     ;Code  Symbo23 '>'
;       .DB     $08,$14,$22,$41,$00,$00,$00,$00     ;Code  Symbo24 '<'
;       .DB     $20,$40,$45,$48,$30,$00,$00,$00     ;Code  Symbo25 '?'
;       .DB     $26,$49,$4F,$41,$3E,$00,$00,$00     ;Code  Symbo26 '@'
;       .DB     $80,$80,$79,$00,$00,$00,$00,$00     ;Code  Symbo27 '!'
;       .DB     $14,$7F,$14,$7F,$14,$00,$00,$00     ;Code  Symbo28 '#'
;       .DB     $12,$2A,$7F,$2A,$24,$00,$00,$00     ;Code  Symbo29 '$'
;       .DB     $10,$20,$40,$20,$10,$00,$00,$00     ;Code  Symbo30 '^'
;       .DB     $80,$03,$03,$00,$00,$00,$00,$00     ;Code  Symbo31 '.'
;       .DB     $80,$36,$36,$00,$00,$00,$00,$00     ;Code  Symbo32 ':'
```

```
;       .DB     $80,$35,$36,$00,$00,$00,$00,$00     ;Code  Symbo33 ';'
;       .DB     $80,$70,$80,$70,$00,$00,$00,$00     ;Code  Symbo34 '"'
;       .DB     $80,$05,$06,$00,$00,$00,$00,$00     ;Code  Symbo35 ','
;       .DB     $08,$08,$2A,$1C,$08,$00,$00,$00     ;Code  Symbo36 '->'
;       .DB     $08,$1C,$2A,$08,$08,$00,$00,$00     ;Code  Symbo37 '<-'
;       .DB     $FF,$FF,$FF,$FF,$FF,$00,$00,$00     ;Code  Symbo38 '|*|'
;       .DB     $FF,$41,$41,$41,$FF,$00,$00,$00     ;Code  Symbo39 '| |'
```