```
//*****************************************************************************

#define MX_ARD

#define MX_CAL_ARD

#define MX_CLK_SPEED 16000000

#define FCP_NULL Unconnected_Port

#define MX_UART_ID
#define MX_UART_UCSRC

#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <avr\io.h>
#include <avr\interrupt.h>
#include <avr\eeprom.h>
#include <avr\wdt.h>



/*=============================================================================*\
  Use :Include the type definitions
\*=============================================================================*/
#include "C:\ProgramData\MatrixTSL\FlowcodeV8\CAL\internals.c"




/*=============================================================================*\
  Use :panel
     :Variable declarations
     :Macro function declarations
\*=============================================================================*/
#define FCV_FALSE (0)
#define FCV_TRUE (1)
MX_GLOBAL MX_UINT8 FCV_SECONDI = (0x0); // contatore
MX_GLOBAL MX_UINT8 FCV_DECIMI;
MX_GLOBAL MX_UINT8 FCV_DECINEMINUTI = (0x0);
MX_GLOBAL MX_UINT8 FCV_MINUTI = (0x0);
```

```
/*=====================================================================*\
  Use :lut
     :Variable declarations
     :Macro function declarations
\*=====================================================================*/
#define FCVsz_00fb1_lut__FLOATFIXEDLIST 1
#define FCVsz_00fb1_lut__INTLIST 16
#define FCVsz_00fb1_lut__FLOATLIST 1
#define FCVsz_00fb1_lut__INTFIXEDLIST 1
#define FCD_00fb1_lut__INTLIST(ix) pgm_read_byte(&(FCD_00fb1_lut__INTLIST_LUT[ix]))
ROMARRAY_(MX_UINT8) FCD_00fb1_lut__INTLIST_LUT ROMARRAY_E =
{
// Property added elements
  63, 6, 91, 79, 102, 109, 125, 7, 127, 111, 119, 124, 57, 94, 121, 113
// Dynamically added elements

};


/*=====================================================================*\
  Use :led_7seg_quad1
     :Variable declarations
     :Macro function declarations
\*=====================================================================*/
void FCD_0fca1_led_7seg_quad1__ShowDigit(MX_UINT8 FCL_DIGIT, MX_UINT8 FCL_VALUE,
MX_UINT8 FCL_DECIMALPOINT);

/*=====================================================================*\
  Use :Include the chip adaption layer
\*=====================================================================*/
#include "C:\ProgramData\MatrixTSL\FlowcodeV8\CAL\includes.c"


/*=====================================================================*\
  Use :lut
     :Macro implementations
\*=====================================================================*/

/*=====================================================================*\
  Use :led_7seg_quad1
     :Macro implementations
```

```
\*=========================================================================*/
/*=-------------------------------------------------------------------=*\
  Use :Set the number and decimal point to be displayed in the given digit of the display.
      :
      :Parameters for macro ShowDigit:
      :  Digit : Which of the four digits to change. (0...3)
      :  Value : The number value to set the digit to (0...15), 16=Clear
      :  DecimalPoint : Whether to show the decimal point for the chosen digit.
\*=-------------------------------------------------------------------=*/
void FCD_0fca1_led_7seg_quad1__ShowDigit(MX_UINT8 FCL_DIGIT, MX_UINT8 FCL_VALUE,
MX_UINT8 FCL_DECIMALPOINT)
{

   // Common_Pin0 = Display_Type
   // Common_Pin1 = Display_Type
   // Common_Pin2 = Display_Type
   // Common_Pin3 = Display_Type
   SET_PORT_PIN(A, 0, 0);
   SET_PORT_PIN(A, 1, 0);
   SET_PORT_PIN(A, 2, 0);
   SET_PORT_PIN(A, 3, 0);

   //Comment:
   //Configure the segments

   if (FCL_VALUE < 16)
   {

      FCL_VALUE = FCD_00fb1_lut__INTLIST(FCL_VALUE);

      // pin0 = Display_Type == (.Value & 1)
      // .Value = .Value >> 1
      // pin1 = Display_Type == (.Value & 1)
      // .Value = .Value >> 1
      // pin2 = Display_Type == (.Value & 1)
      // .Value = .Value >> 1
      // pin3 = Display_Type == (.Value & 1)
      // .Value = .Value >> 1
      // pin4 = Display_Type == (.Value & 1)
      // .Value = .Value >> 1
      // pin5 = Display_Type == (.Value & 1)
      // .Value = .Value >> 1
```

```
    // pin6 = Display_Type == (.Value & 1)
    SET_PORT_PIN(B, 0, 0 == (FCL_VALUE & 1));
    FCL_VALUE = FCL_VALUE >> 1;
    SET_PORT_PIN(B, 1, 0 == (FCL_VALUE & 1));
    FCL_VALUE = FCL_VALUE >> 1;
    SET_PORT_PIN(B, 2, 0 == (FCL_VALUE & 1));
    FCL_VALUE = FCL_VALUE >> 1;
    SET_PORT_PIN(B, 3, 0 == (FCL_VALUE & 1));
    FCL_VALUE = FCL_VALUE >> 1;
    SET_PORT_PIN(B, 4, 0 == (FCL_VALUE & 1));
    FCL_VALUE = FCL_VALUE >> 1;
    SET_PORT_PIN(B, 5, 0 == (FCL_VALUE & 1));
    FCL_VALUE = FCL_VALUE >> 1;
    SET_PORT_PIN(B, 6, 0 == (FCL_VALUE & 1));

    // Pin7 = Display_Type == (.DecimalPoint & 1)
    SET_PORT_PIN(B, 7, 0 == (FCL_DECIMALPOINT & 1));

} else {

    if (FCL_VALUE == 16)
    {

        // pin0 = Display_Type == 0
        // pin1 = Display_Type == 0
        // pin2 = Display_Type == 0
        // pin3 = Display_Type == 0
        // pin4 = Display_Type == 0
        // pin5 = Display_Type == 0
        // pin6 = Display_Type == 0
        SET_PORT_PIN(B, 0, 0 == 0);
        SET_PORT_PIN(B, 1, 0 == 0);
        SET_PORT_PIN(B, 2, 0 == 0);
        SET_PORT_PIN(B, 3, 0 == 0);
        SET_PORT_PIN(B, 4, 0 == 0);
        SET_PORT_PIN(B, 5, 0 == 0);
        SET_PORT_PIN(B, 6, 0 == 0);

    // } else {

    }
```

```
  }

  switch (FCL_DIGIT)
  {
    case 1:
    {
      // Common_Pin1 = 1 - Display_Type
      SET_PORT_PIN(A, 1, 1 - 0);

      break;
    }
    case 2:
    {
      // Common_Pin2 = 1 - Display_Type
      SET_PORT_PIN(A, 2, 1 - 0);

      break;
    }
    case 3:
    {
      // Common_Pin3 = 1 - Display_Type
      SET_PORT_PIN(A, 3, 1 - 0);

      break;
    }
    default:
    {
      // Common_Pin0 = 1 - Display_Type
      SET_PORT_PIN(A, 0, 1 - 0);

    }
  }

}


/*=========================================================================*\
  Use :panel
    :Macro implementations
\*=========================================================================*/
```

```
/*================================================================================*\
  Use :Main
\*================================================================================*/
int main()
{
   MCUSR=0x00;


   // Name: Loop, Type: Loop: While 1
   while (1)
   {

      // Name: Call Component Macro, Type: Call Component Macro:
led_7seg_quad1::ShowDigit(3, secondi, 0)
      FCD_0fca1_led_7seg_quad1__ShowDigit(3, FCV_SECONDI, 0);

      // Name: Call Component Macro, Type: Call Component Macro:
led_7seg_quad1::ShowDigit(2, decimi, 0)
      FCD_0fca1_led_7seg_quad1__ShowDigit(2, FCV_DECIMI, 0);

      // Name: Call Macro, Type: Call Component Macro: led_7seg_quad1::ShowDigit(1, minuti, 1)
      FCD_0fca1_led_7seg_quad1__ShowDigit(1, FCV_MINUTI, 1);

      // Name: Call Macro, Type: Call Component Macro: led_7seg_quad1::ShowDigit(0,
decineminuti, 0)
      FCD_0fca1_led_7seg_quad1__ShowDigit(0, FCV_DECINEMINUTI, 0);

      // Name: Calculation, Type: Calculation:
      // secondi = secondi + 1
      FCV_SECONDI = FCV_SECONDI + 1;

      // Name: Decision, Type: Decision: secondi > 9?
      if (FCV_SECONDI > 9)
      {

         // Name: Calculation, Type: Calculation:
         // secondi = 0
         // decimi = decimi + 1
         // decimi = decimi % 6
         FCV_SECONDI = 0;
         FCV_DECIMI = FCV_DECIMI + 1;
         FCV_DECIMI = FCV_DECIMI % 6;
```

```
        // Name: Decision, Type: Decision: decimi = 0?
        if (FCV_DECIMI == 0)
        {

          // Name: Calculation, Type: Calculation:
          // minuti = minuti + 1
          // minuti = minuti % 10
          FCV_MINUTI = FCV_MINUTI + 1;
          FCV_MINUTI = FCV_MINUTI % 10;

          // Name: Decision, Type: Decision: minuti = 0?
          if (FCV_MINUTI == 0)
          {

            // Name: Calculation, Type: Calculation:
            // decineminuti = decineminuti + 1
            // decineminuti = decineminuti % 6
            FCV_DECINEMINUTI = FCV_DECINEMINUTI + 1;
            FCV_DECINEMINUTI = FCV_DECINEMINUTI % 6;

          // } else {

          }

        // } else {

        }

      // } else {

      }

      // Name: Delay, Type: Delay: 100 ms
      FCI_DELAYBYTE_MS(100);


    }

    mainendloop: goto mainendloop;
}
```

```
/*====================================================================*\
  Use :Interrupt
\*====================================================================*/
```