MGL-based IP Core: Alphanumeric Display Driver

Features

- **Alphanumeric Display Macro**
- **Easy to Implement on AT94K FPGA**
- **Display on Alphanumeric Display of the ATSTK94 Starter Kit**

Introduction

This alphanumeric display macro is implemented by a 4-input Look-UP Table (LUT) with 15 outputs/segments. Each output/segment of the alphanumeric display is a small light-emitting diode (LED) or liquid-crystal display (LCD). A hexadecimal number is indicated by lighting a particular combination of the LED's elements.

Description

This application note creates a design that takes the 4-bit input binary number and displays it by lighting the corresponding segments on the alphanumeric display.

Download **3021.zip** to obtain the code for this application note.

The alphanumeric display is shown in Figure 1. The number 0 is displayed by lighting the segments a, b, c, d, e, and f, while the segment g, h, j, k, l, m, n, p, and dp are unlit. The four inputs are represented as binary variables A, B, C, and D[1], see Figure 2.

Thus if A = B = C = D = 0, which means the input number is (0000)binary or 0 in decimal, then the outputs a = b = c = d = e = f = 1 and g = h = j = k = l = m = n = p = dp = 0, where 1 means lit and 0 means unlit.

Note: 1. Since *A, B, C* and *D* can only represent numbers from 0 to F in hexadecimal, *dp* is not used in this particular example.

Programmable SLI
AT94K
AT94S

Application Note

Rev. 3021A–FPSLI–06/02

Figure 1. Alphanumeric Display

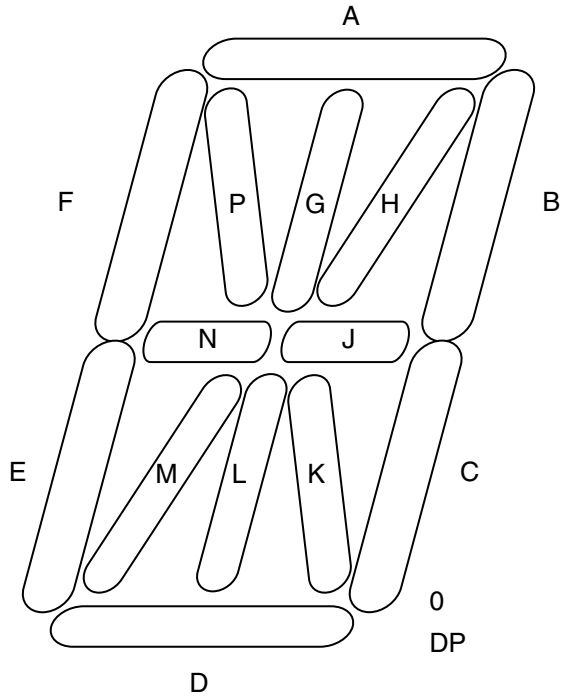
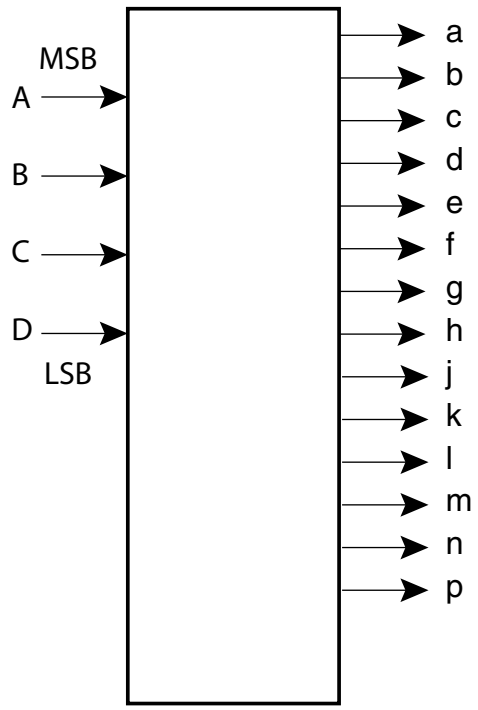


Figure 2. Alphanumeric Display – I/Os



As another example, if (ABCD) = (0010), the resulting display will be 2. From the above design specifications, it is easy to construct the truth table of the alphanumeric display. Note that the largest binary input number is 1111 = F in hexadecimal, see Table 1 on page 3.

Table 1. Alphanumeric Truth Table

	dp	p	n	m	l	k	j	h	g	f	e	d	c	b	a
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
2	0	0	1	0	0	0	1	0	0	0	1	1	0	1	1
3	0	0	1	0	0	1	0	1	0	0	0	1	0	0	1
4	0	0	1	0	1	0	0	0	1	1	0	0	0	0	0
5	0	0	1	0	0	0	1	0	0	1	0	1	1	0	1
6	0	0	1	0	0	0	1	0	0	1	1	1	1	0	1
7	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1
8	0	0	1	0	0	0	1	0	0	1	1	1	1	1	1
9	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1
a	0	0	1	0	0	0	1	0	0	0	1	1	1	1	1
b	0	0	1	0	0	0	1	0	0	1	1	1	1	0	0
c	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0
d	0	0	1	0	0	0	1	0	0	0	1	1	1	1	0
e	0	0	1	0	0	0	1	0	0	1	1	1	0	1	1
f	0	0	1	0	0	0	0	0	0	1	1	0	0	0	1

Then the minterm list for a, b, c, …, dp will be written from the above true table for each segment.

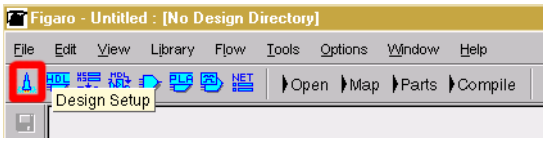
Adding a Macro to the Macro Library

Prior to implementing the design you need to set up the design and add the macro to the macro library.

All the MGL IP cores have a ***.mgl** extension. MGL IP Cores only support VHDL/Verilog designs.

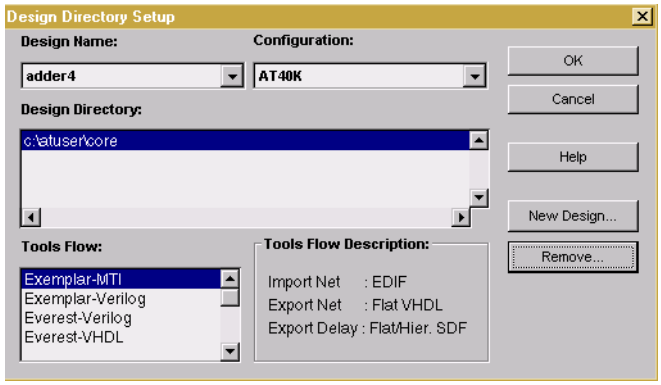
1. Open IDS (Figaro) and click on the Design Setup button, or select **Design Setup...** from the **File** menu, see Figure 3.

Figure 3. IDS Design Setup Button



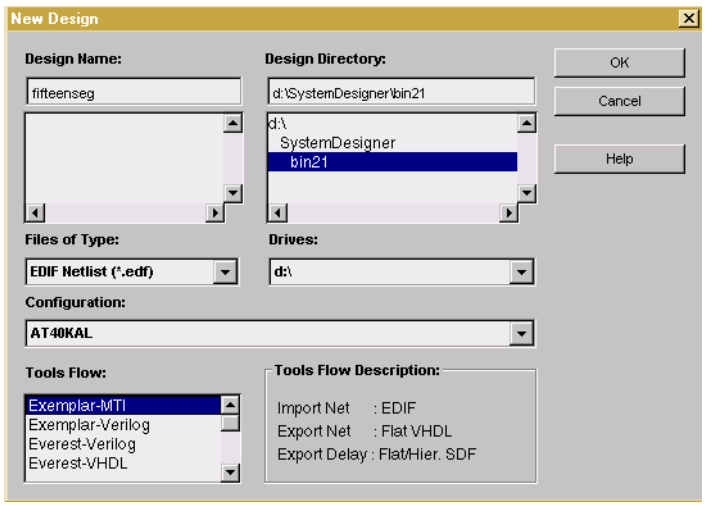
The **Design Directory Setup** dialog box with the last settings appears, see Figure 4.

Figure 4. Design Directory Setup Dialog Box



2. Click on the **New Design...** button. The **New Design** dialog box appears, see Figure 5.

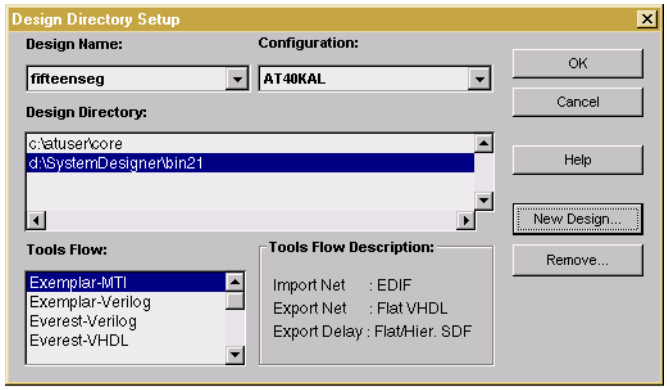
Figure 5. New Design Dialog Box



3. Change the following settings:
   – Type **`fifteenseg.edf`** as **Design Name**.
   – Select the **Design Directory** and **Drive** where you want to save the design.
   – Select **EDIF Netlist (*edf)** from the **Files of Type** list.
   – Select your device from the **Configuration** list.
   – Select the design tool from the **Tools Flow** list. The Tools Flow Description provides the related Import/Export file types.

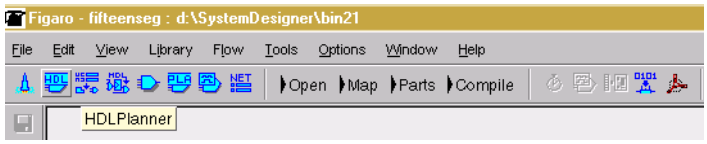
     Note: This version of the software only supports ***.VHDL** Export Net files.
4. Press **OK**. The **Design Directory Setup** dialog box with the current settings appears, see Figure 6.

Figure 6. Design Directory Setup Dialog Box



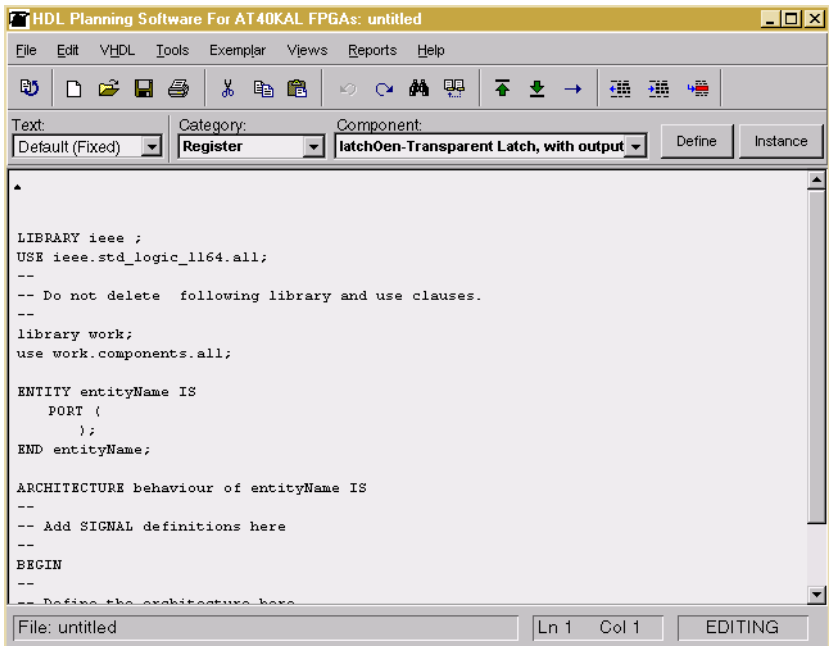
5. Press **OK**.
6. Press the HDLPlanner button, see Figure 7, or select **HDLPlanner** from the **Tools** menu.

Figure 7. HDLPlanner Button



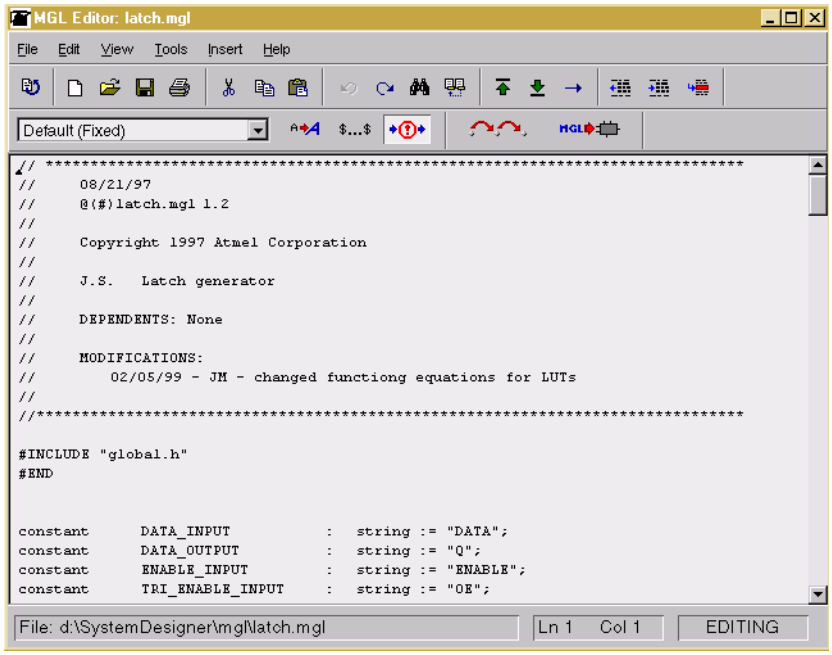
HDLPlanner is invoked, see Figure 8 on page 6.

Figure 8. HDLPlanner Window



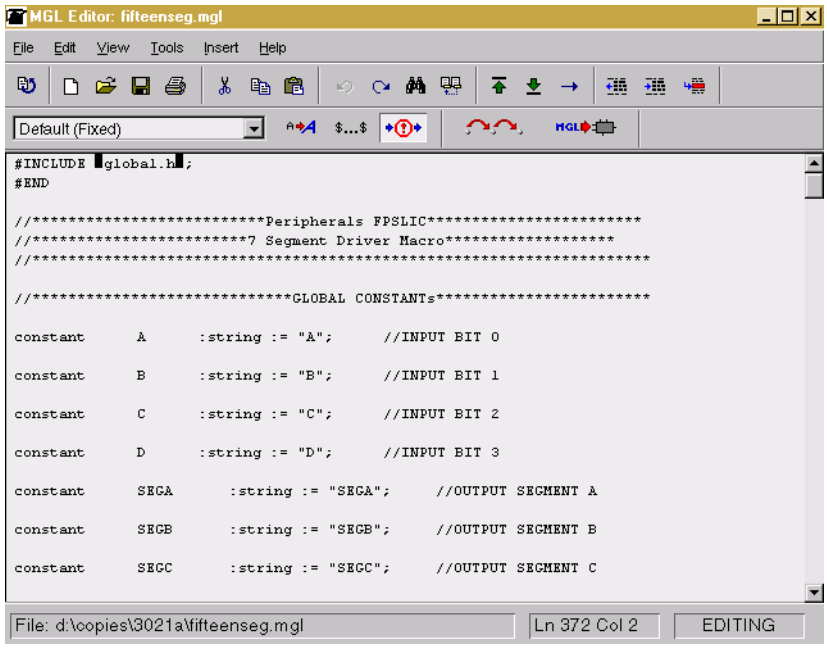
7. Select **Layout...** from the **View** menu. The NMGL editor is invoked, see Figure 9.

Figure 9. Layout View



8. Open the MGL IP Core file **fifteenseg.mgl**, see Figure 10.

Figure 10. FIFTEEN.MGL IP Core File



9. Click the execute MGL code button, see Figure 11, or press **F5** to Compile the file or select **Compile** from the **Tools** menu. The MGL code is now compiled.

   A dialog box indicating a successful compilation appears, see Figure 12.

Figure 11. Execute MGL Button

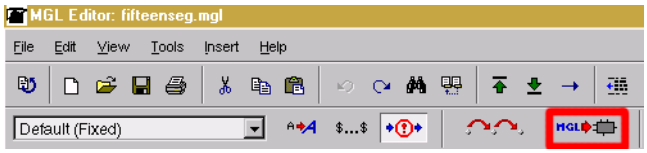
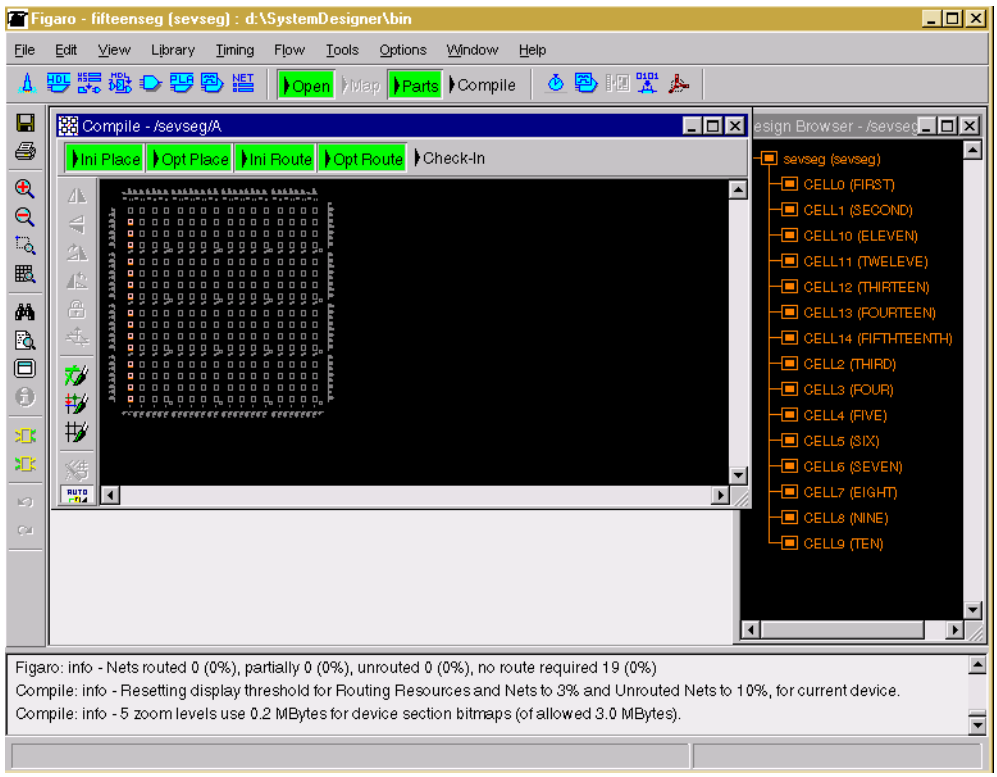


Figure 12. Successful Compilation Dialog Box

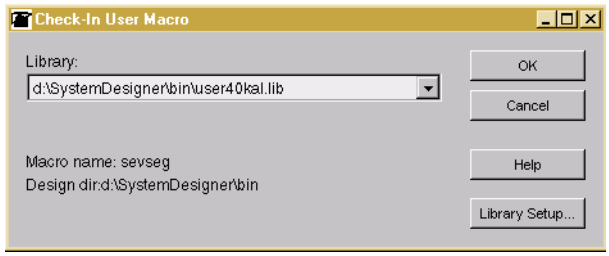
10. Press the **Import Generated Macro into IDS Tool** button. Figaro opens the macro, see Figure 13.

Figure 13. Fifteenseg Macro



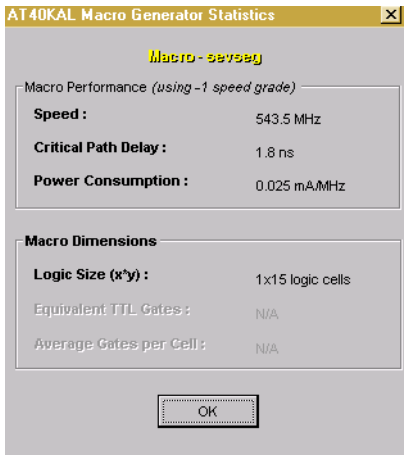
11. Click on the **Check-In** button to check in the macro. The Check-in User Macro dialog box appears, see Figure 14.

Figure 14. Check-in User Macro Dialog Box



12. Select the user library where you want to save the design and click **OK**. The Macro Generator Statistics window appears, see Figure 15.

Figure 15. Macro Generator Statistic Window



13. Press OK. The macro[1] has been successfully checked in.

Note: 1. The macro name can be changed at the bottom of the page. Refer to **mgl_lang.hlp** for more information.

```
begin
    return(aSev( "fifteenseg", 15));
end;
```

Design Implementation

Now you can select the macro from your macro library within your design. Make sure to define the interconnects and assign the I/O pins before running the macro.

Code

The alphanumeric display macro can be modified to display uppercase or expand the letters to “Z”. After making any changes, save and recompile the macro and the design. See the alphanumeric display macro below for the complete source code:

```
#INCLUDE “global.h”;
#END

//**************************Peripherals FPSLIC**********************
//************************7 Segment Driver Macro*******************
//*****************************************************************

//*****************************GLOBAL CONSTANTs**********************

constantA:string := "A";//INPUT BIT 0

constantB:string := "B";//INPUT BIT 1

constantC:string := "C";//INPUT BIT 2

constantD:string := "D";//INPUT BIT 3

constantSEGA:string := "SEGA";     //OUTPUT SEGMENT A

constantSEGB:string := "SEGB";     //OUTPUT SEGMENT B

constantSEGC:string := "SEGC";     //OUTPUT SEGMENT C

constantSEGD:string := "SEGD";     //OUTPUT SEGMENT D

constantSEGE:string := "SEGE";     //OUTPUT SEGMENT E

constantSEGF:string := "SEGF";     //OUTPUT SEGMENT F

constantSEGG:string := "SEGG";     //OUTPUT SEGMENT G

constantSEGH:string := "SEGH";     //OUTPUT SEGMENT H

constantSEGJ:string := "SEGJ";     //OUTPUT SEGMENT J

constantSEGK:string := "SEGK";     //OUTPUT SEGMENT K

constantSEGL:string := "SEGL";     //OUTPUT SEGMENT L

constantSEGM:string := "SEGM";     //OUTPUT SEGMENT M

constantSEGN:string := "SEGN";    //OUTPUT SEGMENT N
```

```
constantSEGP:string := "SEGP";    //OUTPUT SEGMENT P

constantSEGDP:string := "SEGDP";   //OUTPUT SEGMENT DP


function sevCell(firstCell, secondCell, thirdCell, fourthCell, fivthCell,
sixthCell, seventhCell, eighthCell, ninethCell, tenthCell, eleventhCell,
twleveCell, thirteenthCell, fourteenthCell, fifthteenthCell:boolean
         ):macro
      sevCell:macro;
      fgen  :macro;
      interfaceName:string;

begin
      if (firstCell) then
          interfaceName:= "FIRST";
      elseif (secondCell) then
          interfaceName:= "SECOND";
      elseif(thirdCell) then
          interfaceName:= "THIRD";
      elseif (fourthCell) then
          interfaceName:= "FOUR";
      elseif (fivthCell) then
          interfaceName:= "FIVE";
      elseif (sixthCell) then
          interfaceName:= "SIX";
      elseif (seventhCell) then
          interfaceName:= "SEVEN";
      elseif (eighthCell) then
          interfaceName:= "EIGHT";
      elseif (ninethCell) then
          interfaceName:= "NINE";
      elseif (tenthCell) then
          interfaceName:= "TEN";
      elseif (eleventhCell) then
          interfaceName:= "ELEVEN";
      elseif (twleveCell) then
          interfaceName:= "TWELEVE";
      elseif (thirteenthCell) then
          interfaceName:= "THIRTEEN";
      elseif (fourteenthCell) then
          interfaceName:= "FOURTEEN";

      else
          interfaceName:= "FIFTHTEENTH";
      end if;

      interface interfaceName of sevCell is
```

```
        inputports("A");
        inputports("B");
        inputports("C");
        inputports("D");
        outputports("SEGA");
        outputports("SEGB");
        outputports("SEGC");
        outputports("SEGD");
        outputports("SEGE");
        outputports("SEGF");
        outputports("SEGG");
        outputports("SEGH");
        outputports("SEGJ");
        outputports("SEGK");
        outputports("SEGL");
        outputports("SEGM");
        outputports("SEGN");
        outputports("SEGP");
        outputports("SEGDP");

      end interface;

      contents of sevCell is

        fgen := getmacro("FGEN1");

      instance "SEVINST" of fgen is
        location(0, 0);

        if (firstCell) then

functiong("(!A&!B&!C&!D)+(!A&!B&C&!D)+(!A&!B&C&D)+(!A&B&!C&D)+(!A&B&C&!D)+(!
A&B&C&D)+(A&!B&!C&!D)+(A&!B&!C&D)+(A&!B&C&!D)+(A&B&C&!D)+(A&B&C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGA);

        elseif (secondCell) then

functiong("(!A&!B&!C&!D)+(!A&!B&!C&D)+(!A&!B&C&!D)+(A&!B&!C&!D)+(A&!B&!C&D)+
(A&!B&C&!D)+(A&B&!C&D)+(A&B&C&!D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGB);

        elseif (thirdCell) then

functiong("(!A&!B&!C&!D)+(!A&!B&!C&D)+(!A&B&!C&D)+(!A&B&C&!D)+(A&!B&!C&!D)+(
A&!B&!C&D)+(A&!B&C&!D)+(A&!B&C&D)+(A&B&!C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGC);

        elseif (fourthCell) then
```

```
functiong("(!A&!B&!C&!D)+(!A&!B&C&!D)+(!A&!B&C&D)+(!A&B&!C&D)+(!A&B&C&!D)+(A
&!B&!C&!D)+(A&!B&C&!D)+(A&!B&C&D)+(A&B&!C&!D)+(A&B&!C&D)+(A&B&C&!D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGD);

        elseif (fivthCell) then

functiong("(!A&!B&!C&!D)+(!A&!B&C&!D)+(!A&B&C&!D)+(A&!B&!C&!D)+(A&!B&C&!D)+(
A&!B&C&D)+(A&B&!C&!D)+(A&B&!C&D)+(A&B&C&!D)+(A&B&C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGE);

        elseif (sixthCell) then

functiong("(!A&!B&!C&!D)+(!A&B&!C&!D)+(!A&B&!C&D)+(!A&B&C&!D)+(A&!B&!C&!D)+(
A&!B&C&D)+(A&B&C&!D)+(A&B&C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGF);

        elseif (seventhCell) then
          functiong("(!A&B&!C&!D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGG);

        elseif (eighthCell) then
          functiong("(!A&!B&C&D)+(!A&B&C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGH);

        elseif (ninethCell) then

functiong("(!A&!B&C&!D)+(!A&B&!C&D)+(!A&B&C&!D)+(A&!B&!C&!D)+(A&!B&!C&D)+(A&
!B&C&!D)+(A&!B&C&D)+(A&B&!C&!D)+(A&B&!C&D)+(A&B&C&!D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGJ);

        elseif (tenthCell) then
          functiong("(!A&!B&C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGK);

        elseif (eleventhCell) then
          functiong("(!A&B&!C&!D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGL);

        elseif (twleveCell) then
          functiong("(!A&B&C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGM);
```

```
        elseif (thirteenthCell) then

functiong("(!A&!B&C&!D)+(!A&!B&C&D)+(!A&B&!C&!D)+(!A&B&!C&D)+(!A&B&C&!D)+(A&
!B&!C&!D)+(A&!B&C&!D)+(A&!B&C&D)+(A&B&!C&!D)+(A&B&!C&D)+(A&B&C&!D)+(A&B&C&D)
");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGN);

        elseif (fourteenthCell) then
          functiong("(A&!B&!C&D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGP);
        else
          functiong("(!A&!B&!C&!D)");
          connections("A"->A, "B"->B, "C"->C, "D"->D);
          connections("G"-> SEGDP);

        end if;
      end instance;
    end contents;
    return(sevCell);
  end;

  function aSev(name:string,
        width:integer
          ) :macro

        Sev :macro;
        aCell:macro;

  begin
    if (width > 15) then
      error( "Sevseg has only 8 segment" );
    elseif (width < 15 ) then
      error( "Sevseg has only 8 segment" );
    end if;

    interface name of Sev is
        inputports(A, B, C, D);
      for i in 0 to width-1 loop
        if (i=0) then
        outputports(SEGA);
        end if;

        if (i=1) then

        outputports(SEGB);
        end if;
        if (i=2) then
```

```
outputports(SEGC);
end if;
if (i=3) then

outputports(SEGD);
end if;
if (i=4) then

outputports(SEGE);
end if;
if (i=5) then

outputports(SEGF);
end if;
if (i=6) then

outputports(SEGG);
end if;
if (i=7) then

outputports(SEGH);
end if;
if (i=8) then

outputports(SEGJ);
end if;
if (i=9) then

outputports(SEGK);
end if;
if (i=10) then

outputports(SEGL);
end if;
if (i=11) then

outputports(SEGM);
end if;
if (i=12) then

outputports(SEGN);
end if;
if (i=13) then

outputports(SEGP);
end if;
if (i=14) then

outputports(SEGDP);
end if;
```

```
      end loop;

    end interface;

    contents of Sev is
      for i in 0 to width-1 loop
      aCell:= sevCell( i = 0, i=1, i=2, i=3, i=4, i=5, i=6,
i=7,i=8,i=9,i=10,i=11,i=12,i=13,i=width-1);
        instance "CELL"{i} of aCell is
          location(0, i);
          connections(SEGA->SEGA);
          connections(A->A, B->B, C->C, D->D);
          if (i=0) then
          placeports(SEGA->"Y");
          end if;
          if (i =1) then
          placeports(SEGB->"Y");
          connections(SEGB->SEGB);
          end if;
          if (i = 2) then
          placeports(SEGC->"Y");
          connections(SEGC->SEGC);
          end if;
          if (i = 3) then
          placeports(SEGD->"Y");
          connections(SEGD->SEGD);
          end if;
          if (i = 4) then
          placeports(SEGE->"Y");
          connections(SEGE->SEGE);
          end if;
          if (i = 5) then
          placeports(SEGF->"Y");
          connections(SEGF->SEGF);
          end if;
          if (i=6) then
          placeports(SEGG->"Y");
          connections(SEGG->SEGG);
          end if;
          if (i=7) then
          placeports(SEGH->"Y");
          connections(SEGH->SEGH);
          end if;
          if (i=8) then
          placeports(SEGJ->"Y");
          connections(SEGJ->SEGJ);
          end if;
          if (i=9) then
          placeports(SEGK->"Y");
          connections(SEGK->SEGK);
```

```
            end if;
            if (i=10) then
            placeports(SEGL->"Y");
            connections(SEGL->SEGL);
            end if;
            if (i=11) then
            placeports(SEGM->"Y");
            connections(SEGM->SEGM);
            end if;
            if (i=12) then
            placeports(SEGN->"Y");
            connections(SEGN->SEGN);
            end if;
            if (i=13) then
            placeports(SEGP->"Y");
            connections(SEGP->SEGP);
            end if;
            if (i=14) then
            placeports(SEGDP->"Y");
            connections(SEGDP->SEGDP);
            end if;
          end instance;
        end loop;
      end contents;
      return(Sev);
    end;

//*************************************************************************
********
//PROGRAM BLOCK
//*************************************************************************
*********

begin
    return(aSev( "fifteenseg", 15));
end;
```

Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 487-2600

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
TEL (41) 26-426-5555
FAX (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
TEL (33) 2-40-18-18-18
FAX (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
TEL (33) 4-42-53-60-00
FAX (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL 1(719) 576-3300
FAX 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
TEL (44) 1355-803-000
FAX (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
TEL (49) 71-31-67-0
FAX (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL 1(719) 576-3300
FAX 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
TEL (33) 4-76-58-30-00
FAX (33) 4-76-58-34-80

Atmel Programmable SLI Hotline
(408) 436-4119

Atmel Programmable SLI e-mail
fpslic@atmel.com

FAQ
Available on web site

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

3021A–FPSLI–06/02 xM