AN002

Using PLD as a decoder logic for Z80 Trainer Kit

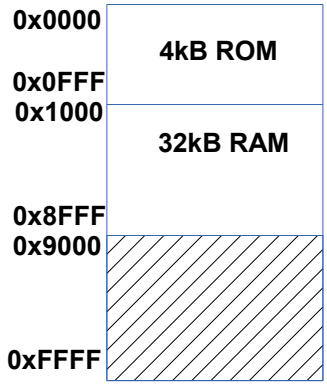
Wichit Sirichote, kswichit@kmitl.ac.th

Introduction

This application note describes the use of PLD as a decoder logic for the memory and I/O of the Z80 Trainer kit. Addresses of memory and I/O space are explained. The equations for memory and I/O are written. Using ATMEL WinCupl compiler to convert logic equations JEDEC file and the example of using G540 device programmer for programming the Lattice GAL16V8 are described.

Memory map

The memory map of total 64kB is shown below. The first boot block (4kB) made with ROM is located at 0x0000 to 0x0FFF. When reset, Z80 will fetch the first byte at reset vector, 0x0000. RAM is placed at 0x1000 to 0x8FFF. The rest from 0x9000 to 0xFFFF is available for memory expansion.



ROM chip enable equation

For location 0x0000 to 0x0FFF (4kB), we see that A15,A14,A13 and A12, must be 0. To select the ROM chip in memory space, we will need MREQ signal. The equation will be,

```
ROMCE = A12 # A13 # A14 # A15 # MREQ;

# is logical OR operator
```

RAM chip enable equation

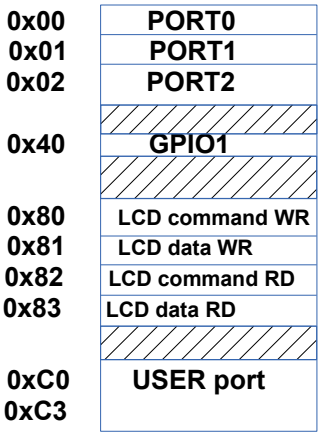
For location 0x1000 to 0x8FFF, we see that A15,A14,A13 and A12, will be 0001 to 1000, or 8 blocks of 4kB each. The equation will be,

```
RAMCE = ((!A12 # A13 # A14#A15)&(A12 # !A13
# A14 # A15)&(!A12 # !A13 # A14 # A15)&(A12
# A13 # !A14 # A15)&(!A12 # A13 # !A14 #
A15) &(A12 # !A13 # !A14 # A15)&(!A12 # !
A13 # !A14 # A15)
&(A12 # A13 # A14 # !A15)) # MREQ;

& is logical AND operator
! is logical NOT operator
```

All blocks enable produced by A15,A14,A13,A12 are AND together and then OR with MREQ signal.

I/O map

The I/O map of total 256 bytes is shown below. For I/O decoder we use only A7,A6 and A1,A0 to select the devices. By using A7 and A6, we can get only 4 blocks and each block has 4 locations which is addressed by A1,A0.



The equations for system Port are,

```
PORT0 = IORQ # A6 # A7 # A0 # A1;

PORT1 = IORQ # A6 # A7 # !A0 # A1;

PORT2 = IORQ # A6 # A7 # A0 # !A1;
```

*PORT3 is available for user.

GPIO1 is designed to display 8-bit binary number.

The address is 0x40.

```
GPIO1 = IORQ # !A6 # A7 # A0 # A1;
```

*Locations 0x41, 0x42 and 0x43 are available for user

For text LCD that interfaces to the Z80 data bus directly, The locations of LCD's register are 0x80 to 0x83. Since the enable signal, is active high, thus the equation will be,

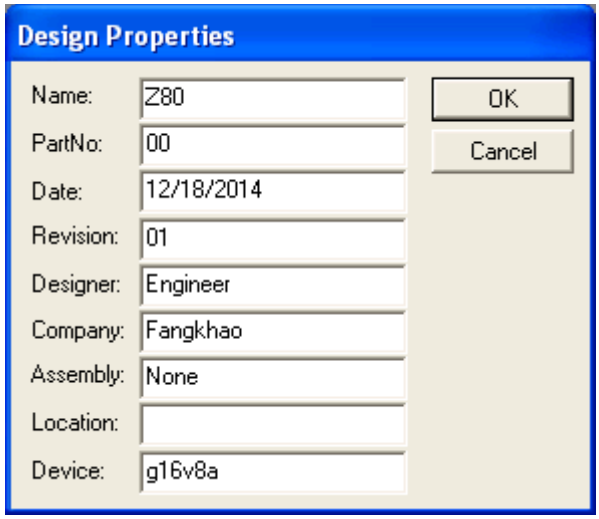
```
!LCD_E = IORQ # A6 # !A7;
```

The rest I/O from 0xC0 to 0xC3 are for expansion.

Using WinCupl

To convert the logical equation to JEDEC file for PLD device, we will need the PLD compiler. The example is ATMEL WinCupl version 5.30.4.

Click file, new deign project, enter name and part number. Important note, at the device, we must enter the PLD name, g16v8a. Otherwise the program will enter virtual device. If the device name is virtual, the compiler will not produce the JEDEC file!

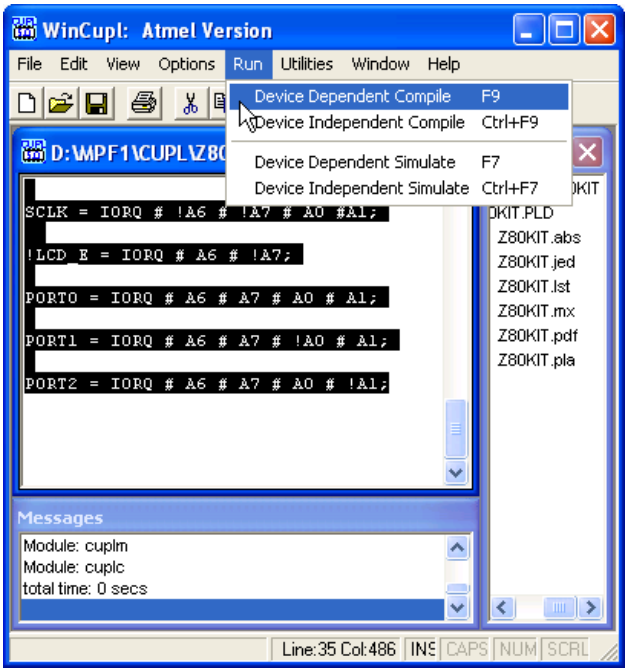


Then edit the designated functions for each pins. And the logical equations.

```
/******* INPUT PINS *****/
PIN 2     =      MREQ;
PIN 3     =      IORQ;
PIN 4     =      A6;
PIN 5     =      A7;
PIN 6     =      A12;
PIN 7     =      A13;
PIN 8     =      A14;
PIN 9     =      A15;
PIN 1     =      A0;
PIN 11     =     A1;

/******* OUTPUT PINS *****/
PIN   12    =    ROMCE;
PIN   13    =    RAMCE;
PIN   14    =    GPIO1;
PIN   15    =    SCLK;
PIN   16    =    LCD_E;
PIN   17    =    PORT0;
PIN   18    =    PORT1;
PIN   19    =    PORT2;

ROMCE = A12 # A13 # A14 # A15 # MREQ;


RAMCE = ((!A12 # A13 # A14#A15)&(A12 # !A13 #
A14 # A15)&(!A12 # !A13 # A14 # A15)&(A12 #
A13 # !A14 # A15)&(!A12 # A13 # !A14 # A15)
&(A12 # !A13 # !A14 # A15)&(!A12 # !A13 # !
A14 # A15)
&(A12 # A13 # A14 # !A15)) # MREQ;

GPIO1 = IORQ # !A6 # A7 # A0 # A1;

!LCD_E = IORQ # A6 # !A7;

PORT0 = IORQ # A6 # A7 # A0 # A1;

PORT1 = IORQ # A6 # A7 # !A0 # A1;

PORT2 = IORQ # A6 # A7 # A0 # !A1;
```

Save project file with PLD extension.

Click Run, Device Dependent Compile or F9.



The message window will show the success of compilation.

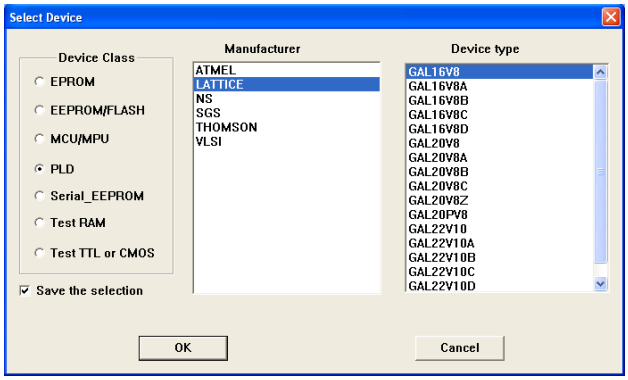
The right window will display the output files. We will use JEDEC file for chip programming. The example produced the Z80KIT.jed. Its content is the fuse map ready for PLD programmer.

```
*L00000 11110111111111111111111111111111
*L00032 11111111011111111111111111111111
*L00064 11111111111101111111111111111111
*L00096 11011111111111111111111111111111
*L00128 11111111111111111111111111111110
*L00256 11110111111111111111111111111111
*L00288 11111111011111111111111111111111
*L00320 11111111111101111111111111111111
*L00352 11101111111111111111111111111111
*L00384 11111111111111111111111111111101
*L00512 11110111111111111111111111111111
*L00544 11111111011111111111111111111111
*L00576 11111111111101111111111111111111
*L00608 11011111111111111111111111111111
*L00640 11111111111111111111111111111101
*L00768 11110111111111111111111111111111
*L00800 11111111011111111111111111111111
*L00832 11111111111101111111111111111111
*L01024 11110111111111111111111111111111
*L01056 11111111011111111111111111111111
*L01088 11111111111101111111111111111111
*L01120 11011111111111111111111111111111
*L01152 11111111111111111111111111111101
*L01280 11110111111111111111111111111111
*L01312 11111111011111111111111111111111
*L01344 11111111111101111111111111111111
*L01376 11011111111111111111111111111111
*L01408 11111111111111111111111111111101
*L01536 11111111111111110111111111110111
*L01568 11111111111111111111011111110111
*L01600 11111111111111111101101110110111
*L01632 11111111111111111111111101110111
*L01664 01111111111111111111111111111111
*L01792 11111111111111111011111111111111
*L01824 11111111111111111111011111111111
*L01856 11111111111111111111111101111111
*L01888 11111111111111111111111111110111
*L01920 01111111111111111111111111111111
*L02048 11101111001100010011011001010110
*L02080 00111000001000000000000000000000
*L02112 00000000000000001111111111111111
*L02144 11111111111111111111111111111111
*L02176 11111111111111110
```

GAL programming

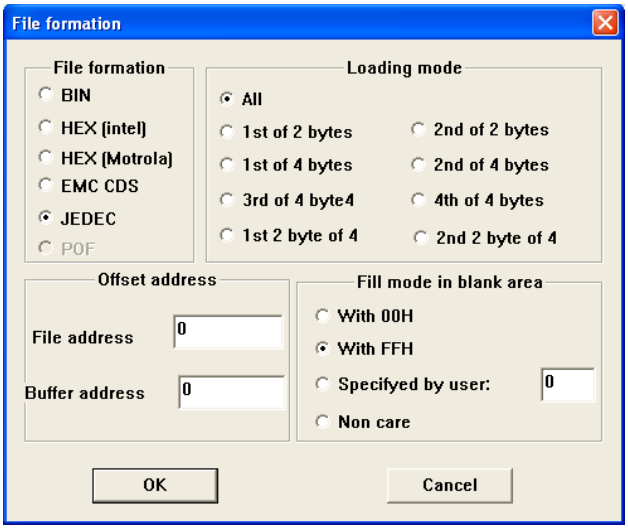
Many PLD programmers accept standard JEDEC file. The example one is G540 Device Programmer. The programmer connects USB port and no need external DC supply.

The Z80 trainer kit uses GAL16V8 PLD chip. So select PLD device and click at Lattice GAL16V8.



The software will show placement orientation. Put the chip to the ZIF socket, lock it and then,

Open the JEDEC file. Our file is Z80KIT.jed.



File formation will shows JEDEC, click OK.

Click Program, the chip will be erased, programmed and verified automatically.

Conclusion

The decoder logic for memory and I/O space can be replaced with a programmable logic device, PLD. The logic equations are written with the relationship between input and output pins. We can use WinCupl PLD compiler to translate the logic equation to JEDEC file.

Resources

1. PLD compiler,
http://www.atmel.com/tools/WINCUPL.aspx

2. G540 Device programmer,
http://www.stg51.com/