

Tutorial Xilinx Virtex-5 FPGA ML506 Edition

Department of Electrical and Computer Engineering Real-Time DSP and FPGA Development Lab

Mark Sison Manalo and Ashkan Ashrafi



Table of Contents

Blank Page	3
Overview	4
About this tutorial	4
Software needed	4
Hardware needed	4
Setting up the ML506 board	5
The ML506	
Connecting the board	5
Preparing the Compact Flash (Only when necessary)	10
Running the Xilinx Demo	14
Prerequisites	
Programming the FPGA with a VHDL Design Using iMPACT - 2 bit greater than circuit	16
Summary	16
The VHDL Design	
Programming the FPGA using iMPACT	
Hardware Co-Sim with a System Generator created Design Using a Black Box - 2-bit greater-	than from
above	34
Summary	34
Creating the Design	
Using Xilinx XPS and SDK to implement serial communication using the RS232 cable and a	
program	
Summary	
Xilinx XPS	
Xilinx SDK	
Play sound with a sine wave using the AC97 codec	
Summary	
Xilinx XPS	
Xilinx SDK	
Program the FPGA	
Programming your FPGA with the Compact Flash	
Summary	
Creating an .ace file	
Setting up the Compact Flash	
Conclusion	
More Designs	114

Blank Page

Overview

About this tutorial

This tutorial will help you familiarize yourself with Xilinx's XtremeDSP Development Platform – Virtex-5 FPGA ML506 Edition. Familiarity of C, VHDL and MATLAB/Simulink would help but it is not required. The ML505/506/507 are the same boards, only the FPGA is different so this tutorial will apply to all of them. Parts of this tutorial are taken from Xilinx tutorials available here:

http://www.xilinx.com/ml506

http://www.xilinx.com/support/documentation/ml506.htm

http://www.xilinx.com/products/boards/ml506/reference_designs.htm

In addition to the Xilinx documents, the following websites and books were also referenced:

http://www.fpgadeveloper.com/

http://myfpgablog.blogspot.com/2009/12/sysgen-create-new-hwcosim-target-with.html http://academic.csuohio.edu/chu_p/rtl/fpga_vhdl.html

Software needed

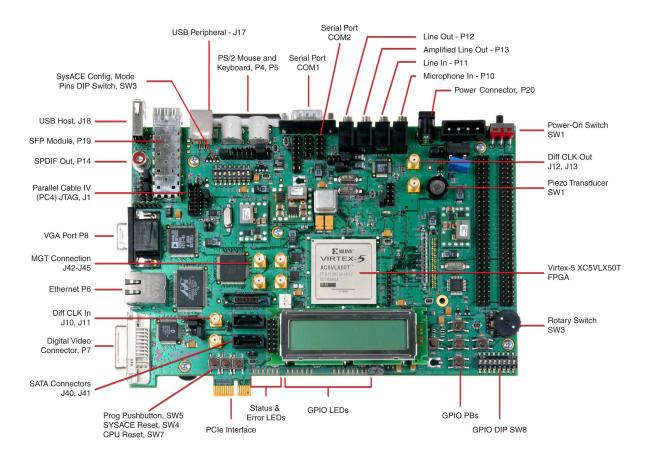
- Xilinx ISE Design Suite (version 12.1 and 13.2 is used here)
- Xilinx Platform Studio
- Xilinx Software Development Kit
- Xilinx System Generator
- MATLAB/SIMULINK
- Hyper Terminal / Tera Term / Putty or similar software (search it on Google and install)
- msdosfs.exe with Windows XP (used for reformatting CF in case of file system corruption)

Hardware needed

- DVI cable or VGA cable with DVI adapter
- RS232 cable for UART
- Xilinx Platform USB cable
- Speakers or headphones
- Compact Flash reader (to load programs into the compact flash)

Setting up the ML506 board

The ML506



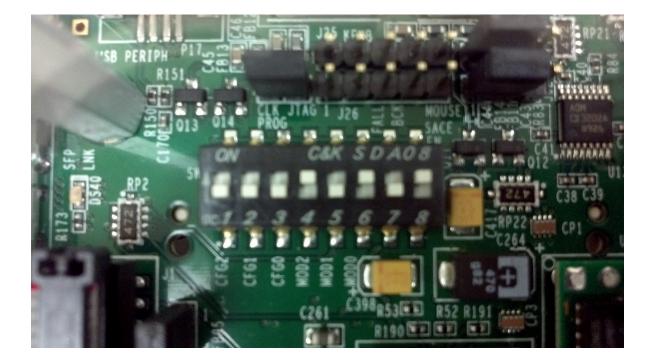
Here is a picture of the ML506 evaluation board with its components labeled.

Connecting the board

The board should already be connected in the FPGA lab. If not, please refer to the Xilinx document titled: ml505_overview_setup.pdf

For convenience, I will summarize it here.

• Set the SW3 switches to 00010101 as shown in the picture:



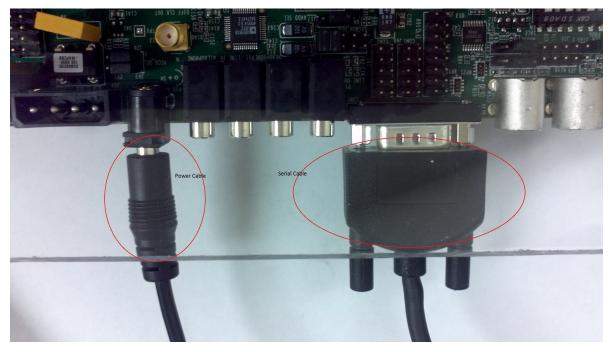
Bits 4-8 (10101) means that on power up, the ML506 board will program the FPGA with the ACE file pointed to by bits 1-3 (in this case: 000). These ACE files are located on the compact flash. 000 means that the ACE file located in cfg0 of the compact flash will be programed into the FPGA. The factory compact flash files has the cfg0 folder containing a system_bootload .ace file. So if you want to load your own ACE file into folder cfg6 (say). Then you would set SW3 to read 110 10101 if you want cfg7 then SW3 = 111 10101 etc... (Search <u>http://www.fpgadeveloper.com/</u> for a tutorial on creating your own ACE file). But for now lets stick the with factory default files.

mputer Removable Disk (F:) ML50X		✓ Search ML5.
re with 🔻 Burn New folder		
Name	Date modified	Type Size
鷆 cfg0	7/6/2011 9:33 AM	File folder
\mu cfg1	7/6/2011 9:33 AM	File folder
\mu cfg2	7/6/2011 9:33 AM	File folder
\mu cfg3	7/6/2011 9:33 AM	File folder
퉬 cfg4	7/6/2011 9:33 AM	File folder
\mu cfg5	7/6/2011 9:33 AM	File folder
\mu cfg6	7/6/2011 9:33 AM	File folder
퉬 cfg7	7/6/2011 9:33 AM	File folder

- Set SW6 (on the back side of the board) to 11001010
- Connect the Xilinx Platform Cable to PC4 JTAG



- Connect the Serial cable (RS232) to the PC either directly or with a serial to USB adapter
- Connect the Power cable to the board



• Connect the DVI cable from the ML506 to a monitor.



• Insert the compact flash into the board.



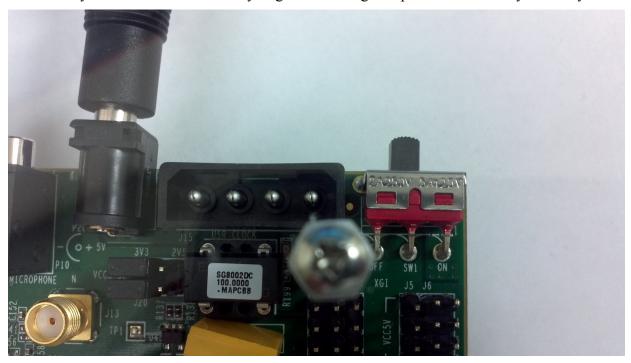
• Install a terminal program such as Tera Term Hyper Terminal or Putty and set it up with the following settings. (On my computer the Serial cable is connected to COM1, yours may differ)

Tera Term: New c	onnection X	
© ТСР <u>/І</u> Р	Hos <u>t</u> : myhost.example.com	
	✓ History Service: ○ Telnet TCP port#: 22	
	O Other Protocol: UNSPEC ▼	
Serial Se	Po <u>r</u> t: COM1: Communications Port (COM1) -	
	OK Cancel <u>H</u> elp	

• Go to setup \rightarrow serial port, and use these settings.

Tera Term: Serial port setup	×
Port:	СОМ1 - ОК
<u>B</u> aud rate:	9600 -
<u>D</u> ata:	8 bit - Cancel
P <u>a</u> rity:	none 🔻
<u>S</u> top:	1 bit • <u>H</u> elp
<u>F</u> low control:	none -
Transmit delay 0 msec <u>k</u>	<u>c</u> har 0 msec/ <u>l</u> ine

Now your board should be ready to go. Don't forget to power it on when your ready.



Preparing the Compact Flash (Only when necessary)

The compact flash should already be formatted correctly as FAT12 or FAT16. After connecting the board insert the compact flash and power up the board. If the CF is not formatted correctly the LED labeled: SACE ERR would be a solid red color as shown in illustration 1.

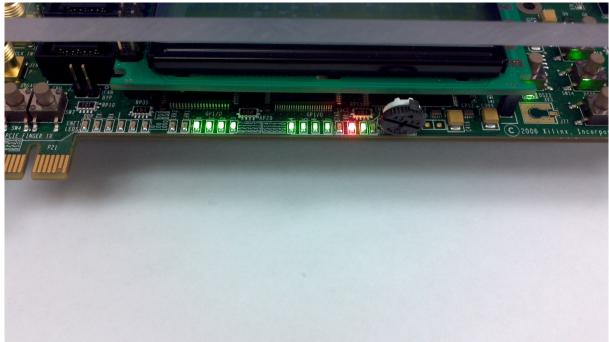


Illustration 1: SACE ERR, CF card is not being read correctly

In this case, one would need to reformat the CF and place the factory default files back into it. Xilinx provides a tutorial on how to do this:

http://www.xilinx.com/products/boards/ml506/ml506_12.1/images.htm

However, my attempts at their tutorial using the dd tool did not work. You can try for yourself if you want. (For the records, I tried this using Windows 7 and I am getting either a non-existent file error or a permissions error, even with administrative privileges)

I managed to reformat the CF by another method, described below. (with the help of this: <u>http://www.xilinx.com/support/answers/14456.htm</u>)

If you are going to reformat the CF, follow the directions carefully. Failure to do so might cause the primary hard drive to be erased .

The following items are required:

- Windows XP (I have not confirmed that it works with Windows 7, you can try though)
- mkdosfs.exe (<u>http://www.plunder.com/MKDOSFS-for-Windows-Format-drives-larger-than-32GB-as-FAT-or-FAT32-download-f738e8fe97.htm</u>)
- Compact flash reader (order at Newegg.com, or some other electronics store)
- ML506 factory CF files, go here: <u>http://www.xilinx.com/products/boards/ml506/ml506_12.1/images.htm</u>



After inserting your CF card into your computer note its drive letter by clicking on Computer.

In my case the drive letter is F: , note that the System ACE controller can only support up to 2GB, the figure above is only for illustrative purposes. The 4GB CF that I have will not work.

Add mkdosfs to your system path: Click start, right-click Computer \rightarrow Properties \rightarrow Advanced system settings \rightarrow Environment Variables. Then edit the 'Path' variable to include the location of where you downloaded mkdosfs.exe. Google how to add to system path for more info.

Then open up a command prompt in windows, preferably with administrative privileges (right click command prompt -> run as administrator).

Type the following command:

mkdosfs -v -F 16 F:

Where 'F:' would be changed to your CF drive letter. Leave the first F as '-F'

After this completes, copy and paste the CF factory files into the compact flash as shown:

Organize ▼ Share with ▼ Burn New fold				•== •	?
🖌 🔆 Favorites	Name	Date modified	Туре	Size	
🧮 Desktop	3 ML50X	7/8/2011 9:38 AM	File folder		
鷆 Downloads	🐌 ringtone	7/8/2011 9:38 AM	File folder		
🕮 Recent Places	📥 demo	10/28/2004 12:13	VLC media file (.bi	12 KB	
Recorded TV	🛃 image01	1/24/2007 10:10 AM	Bitmap image	901 KB	
	🛃 image02	11/17/2006 9:43 AM	Bitmap image	901 KB	
4 🥽 Libraries	🛃 image03	11/17/2006 12:42	Bitmap image	901 KB	
Documents	🛃 image04	11/17/2006 12:41	Bitmap image	901 KB	
🖻 🌙 Music	🛃 image05	11/17/2006 12:41	Bitmap image	901 KB	
Pictures	🛃 image06	11/17/2006 12:41	Bitmap image	901 KB	
Videos	🛃 image07	11/17/2006 12:42	Bitmap image	901 KB	
	🛃 image08	11/17/2006 12:42	Bitmap image	901 KB	
🛛 🔣 Homegroup	🛃 image09	11/17/2006 9:43 AM	Bitmap image	901 KB	
	README	7/7/2006 3:43 PM	Text Document	1 KB	
🛿 📜 Computer	📥 sound	8/27/2004 2:10 PM	VLC media file (.w	5,513 KB	
🛛 🚣 Local Disk (C:)	🚳 xilinx.sys	4/10/2006 8:53 AM	System file	1 KB	
Storage (D:)					
Removable Disk (F:)					
🖻 👝 READYBOOST (J:)					
MB860					

Once this is complete, right-click the compact flash and eject it.

janize 🔻	Share with 🔻 🛛 Burn 🛛 New folder				•
Favorites		Name	Date modified	Туре	Size
Desktop		ML50X	7/8/2011 9:38 AM	File folder	
Downloads		ingtone	7/8/2011 9:38 AM	File folder	
Recent Plac	es	🛓 demo	10/28/2004 12:13	VLC media file (.bi	12
Recorded T	v	image01	1/24/2007 10:10 AM	Bitmap image	901
		🔜 image02	11/17/2006 9:43 AM	Bitmap image	901
Libraries		🔜 image03	11/17/2006 12:42	Bitmap image	901
Documents	5	🛃 image04	11/17/2006 12:41	Bitmap image	901
Music 🖉		🛃 image05	11/17/2006 12:41	Bitmap image	901
Pictures		🛃 image06	11/17/2006 12:41	Bitmap image	901
Videos		🛃 image07	11/17/2006 12:42	Bitmap image	901
		🛃 image08	11/17/2006 12:42	Bitmap image	901
Homegroup		🛃 image09	11/17/2006 9:43 AM	Bitmap image	901
		README	7/7/2006 3:43 PM	Text Document	1
Computer		📥 sound	8/27/2004 2:10 PM	VLC media file (.w	5,513
🍒 Local Disk ((C:)	🚳 xilinx.sys	4/10/2006 8:53 AM	System file	11
Storage (D:)					
Removable					
READY	Expand				
MB860 🦉	Add to archive				
	Open AutoPlay				
Network	Open in new window				
	Share with				
	Open as Portable Device				
	Add to "Archive.rar"				
	Compress and email				
	Compress to "Archive.rar" and email				
	Format				
	Eject				
	Cut				
	Сору				

Insert the CF back into the ML506 and turn on the power. Hopefully, you will have no red SACE ERR.



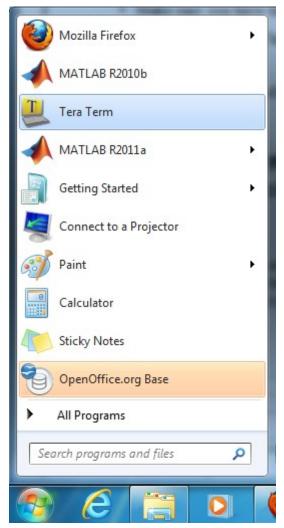
No SACE ERR

Running the Xilinx Demo

Prerequisites

- Make sure you have the board connected to the computer (see—connecting the board)
- Insert the Compact Flash with the factory files loaded into it.
- Connect some headphones or speakers to Line out.

After connecting your board. Run Tera Term/HyperTerminal/Putty with the setting described in the section 'Connecting the Board '.



Power up the board and it should start displaying text to the Tera Term window as shown below.

🐸 COM1:9600baud - Tera Term VT
<u>File Edit Setup Control Window H</u> elp
Helcome to the Xilinx Virtex-5 ML5D5 Evaluation Platform Bootloader Menu!
Please choose a demo by typing in the number of the demo you want to use
Or select a demo using the directional buttons and LCD or VGA display (Then press the center button to start the selected demo)
1. Virtex-5 Slide Shou 2. Heb Server Demo
3. Simon Game
4. Board Diagnostics (XROM) 5. USB Demo
6. My own ACE file
7. Ring Tone Player

You can run the demo programs by pressing a number on your PC keyboard that corresponds to your demo of choice.

Also, this demo makes use of the monitor connected to your ML506 via DVI or VGA adapter. Make sure you switch your screen input (press on the buttons in front of you monitor to access the menu) to the ML506 so you can see the video output.

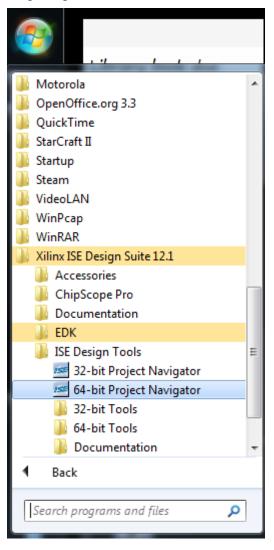
Programming the FPGA with a VHDL Design Using iMPACT – 2 bit greater than circuit

Summary

This tutorial will show you how to program the FPGA with a VHDL design created in ISE. Then we will use 4 switches and 1 led on the ML506 board to simulate the design. We will not go over the VHDL code as it is beyond the scope of this tutorial. After finishing this tutorial, you should be able to program the FPGA with your own circuit design and connect it to some GPIO (general purpose input/output) pins.

The VHDL Design

Open up Xilinx ISE.



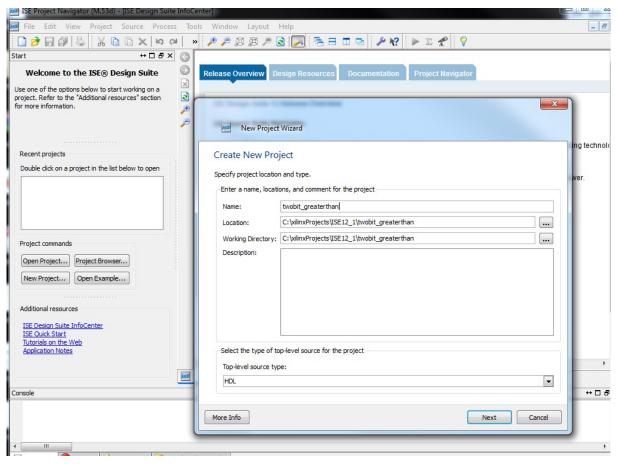
Click File--> New Project.

Set the location of your project file, to avoid any errors later on, make sure there are no spaces in the file path you choose.

Then name your project something descriptive like : twobit greaterthan.

Make sure the top-level source is HDL.

Click Next.



Now setu	n the	project	settings	for the	ML 506	as shown	below a	and cl	ick Next	and Finish.
TNOW SCIU	p inc	project	soungs	ior unc	IVIL JUU	as shown		inu ci	ICK INCAL	and Finish.

Project Settings		
Specify device and project properties.		
Select the device and design flow for the pr	roject	
Property Name	Value	
Product Category	All	[
Family	Virtex5	[
Device	XC5VSX50T	[
Package	FF1136	[
Speed	-1	[
Top-Level Source Type	HDL	
Synthesis Tool	XST (VHDL/Verilog)	[
Simulator	ISim (VHDL/Verilog)	
Preferred Language	VHDL	
Property Specification in Project File	Store non-default values only	[
Manual Compile Order		
VHDL Source Analysis Standard	VHDL-93	[
Enable Message Filtering		

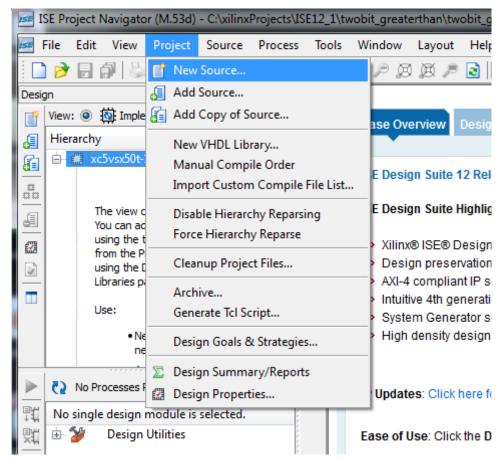
A 2-bit greater-than circuit can be realized as a sum of products, namely,

agreatb = a(1)b(1)' + a(0)b(1)'b(0)' + a(1)a(0)b(0)'

where *a* and *b* are the 2-bit inputs and *agreatb* is the output.

First we will write the VHDL file that implements the circuit, then we will connect inputs a and b to 4 switches and the output agreatb to an led using a universal constraints file.

Highlight the xc5vsx50t in the Hierarchy window, then at the top menu click Project \rightarrow New Source.



Select VHDL module and name the file 'greater_2bit', click Next.

 New Source Wizard Select Source Type Select source type, file name and its location. Select source type, file name and its location. IP (CORE Generator & Architecture Wizard) Schematic User Document Verilog Module Verilog Test Fixture VHDL Module VHDL Library VHDL Package VHDL Test Bench Embedded Processor 	Eile name: greater_2bit Logation: C:\xilinxProjects\ISE12_1\twobit_greaterthan
More Info	Next Cancel

In the define module screen change the architecture name to 'sop_arch' for sum-of-products architecture. We have two inputs, a and b, each two bits wide and one output 'agreatb'. Click Next and Finish.

Define Modu	le					
Specify ports for I	module.					
	greater_2bit					
Architecture name	sop_arch					
	Port Name	Direct	tion	Bus	MSB	LSB
a,b		in	-	V	1	0
agreatb		out	-			
		in	-			
		in	-			
		in	-			
		in	-			
		in	-			
		in	-			
				12 C		

Xilinx should now have created a VHDL template for use to write our code in. If you don't see it double click on the file 'greater_2bit – sop_arch' in the hierarchy window.

Change the architecture definition to this:

```
architecture sop_arch of greater_2bit is
signal p0, p1, p2 : std_logic;
```

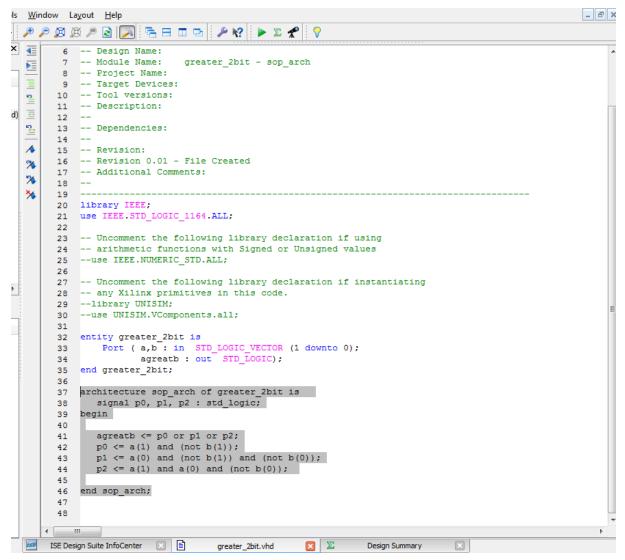
begin

```
agreatb <= p0 or p1 or p2;
p0 <= a(1) and (not b(1));
```

 $p1 \le a(0)$ and (not b(1)) and (not b(0)); $p2 \le a(1)$ and a(0) and (not b(0));

end sop_arch;

Click save.



Now that we have our circuit realized in code, we will connect the inputs and outputs ,a,b,agreatb, to some switches and an led.

To do this we must know the pin numbers of the FPGA. They can be found here:

http://www.xilinx.com/products/boards/ml505/ml505_12.1/docs/ml50x_U1_fpga.ucf

We are interested in these pin locations:

	,,,,,,
NET FPGA VRN B21	LOC="AJ25"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET FPGA VRN B22	LOC="AF8"; # Bank 22, Vcco=3.3V, DCI using 49.9 ohm resistors
NET FPGA VRP B11	LOC="M33"; # Bank 11, Vcco=2.5V or 3.3V user selectable by J20
NET FPGA VRP B13	LOC="AH33"; # Bank 13, Vcco=2.5V or 3.3V user selectable by J20
NET FPGA VRP B17	LOC="AE31"; # Bank 17, Vcco=1.8V, DCI using 49.9 ohm resistors
NET FPGA VRP B19	LOC="M27"; # Bank 19, Vcco=1.8V, DCI using 49.9 ohm resistors
NET FPGA VRP B20	LOC="L11"; # Bank 20, Vcco=3.3V, DCI using 49.9 ohm resistors
NET FPGA VRP B21	LOC="AH25"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET FPGA VRP B22	LOC="AE9"; # Bank 22, Vcco=3.3V, DCI using 49.9 ohm resistors
NET GPIO DIP SW1	LOC="U25"; # Bank 15, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW2	LOC="AG27"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW3	LOC="AF25"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW4	LOC="AF26"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW5	LOC="AE27"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW6	LOC="AE26"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW7	LOC="AC25"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO DIP SW8	LOC="AC24"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO LED 0	LOC="H18"; # Bank 3, Vcco=2.5V, No DCI
NET GPIO LED 1	LOC="L18"; # Bank 3, Vcco=2.5V, No DCI
NET GPIO LED 2	LOC="G15"; # Bank 3, Vcco=2.5V, No DCI
NET GPIO LED 3	LOC="AD26"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO LED 4	LOC="G16"; # Bank 3, Vcco=2.5V, No DCI
NET GPIO LED 5	LOC="AD25"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO LED 6	LOC="AD24"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO LED 7	LOC="AE24"; # Bank 21, Vcco=1.8V, DCI using 49.9 ohm resistors
NET GPIO LED C	LOC="E8"; # Bank 20, Vcco=3.3V, DCI using 49.9 ohm resistors
NET GPIO LED E	LOC="AG23"; # Bank 2, Vcco=3.3V
NET GPIO LED N	LOC="AF13"; # Bank 2, Vcco=3.3V

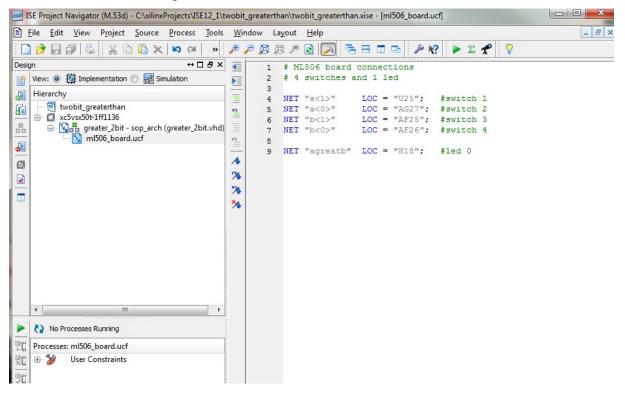
At the top click on Project \rightarrow New Source.

Select Implementation Constraints File and name the file 'ml506_board'. Click Next and finish.

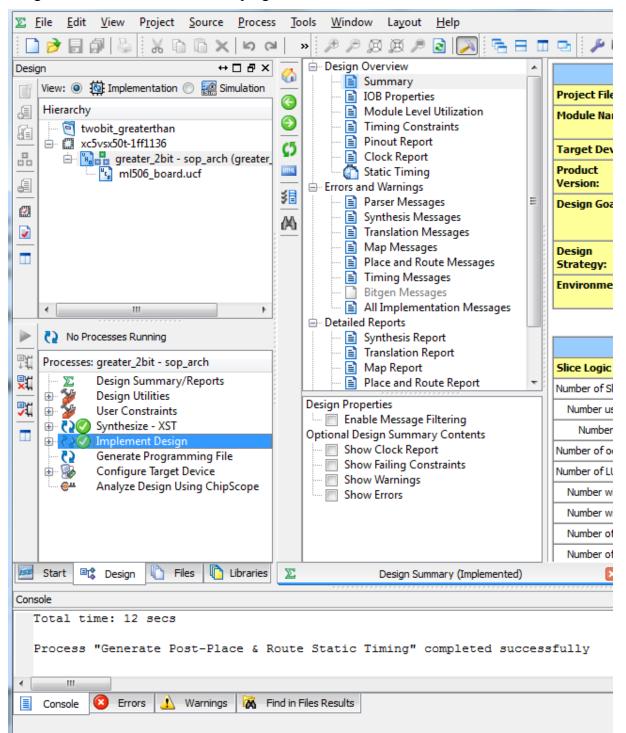
Select Source Type Select source type, file name and its location. BMM File ChipScope Definition and Connection File Timplementation Constraints File Inplementation Constraints File IP (CORE Generator & Architecture Wizard) MEM File Schematic User Document Verilog Module Verilog Test Fixture VHDL Module VHDL Library P VHDL Package VHDL Test Bench Embedded Processor	File name: ml506_board Logation: C:\xilinxProjects\ISE12_1\twobit_greaterthan
More Info	Add to project

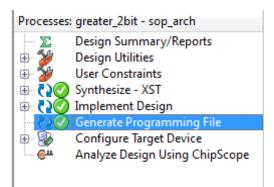
Now you should see a blank file, if not, double-click on 'ml506_board.ucf' in the hierarchy window.

Enter in the following code and save.



Now that our code is finished we have to implement the design and generate the bitstream. Highlight 'greater_2bit -sop_arch' in the Hierarchy window, then in the processes window below it, double-click 'Implement Design' after it finishes double-click 'Generate Programming File', this will generate the bistream needed to program the FPGA.





After Xilinx finishes generating the programming file, you can find it inside your project directory.

퉬 xst	7/15/2011 10:15 AM	File folder
greater_2bit.bgn	7/15/2011 10:19 AM	BGN File
greater_2bit.bit	7/15/2011 10:19 AM	BIT File
greater_2bit.bld	7/15/2011 10:16 AM	BLD File
greater_2bit.cmd_log	7/15/2011 10:19 AM	CMD_LOG File
greater_2bit.drc	7/15/2011 10:19 AM	DRC File
greater_2bit_envsettings	7/15/2011 10:19 AM	Firefox Document
greater_2bit_summary	7/15/2011 10:19 AM	Firefox Document
usage_statistics_webtalk	7/15/2011 10:19 AM	Firefox Document
twobit_greaterthan.gise	7/15/2011 10:19 AM	GISE File
greater_2bit.lso	7/12/2011 11:18 AM	LSO File
areater 2bit map.map	7/15/2011 10:16 AM	MAP File

Programming the FPGA using iMPACT

Now its time to program the FPGA. Close Xilinx ISE and open up iMPACT.

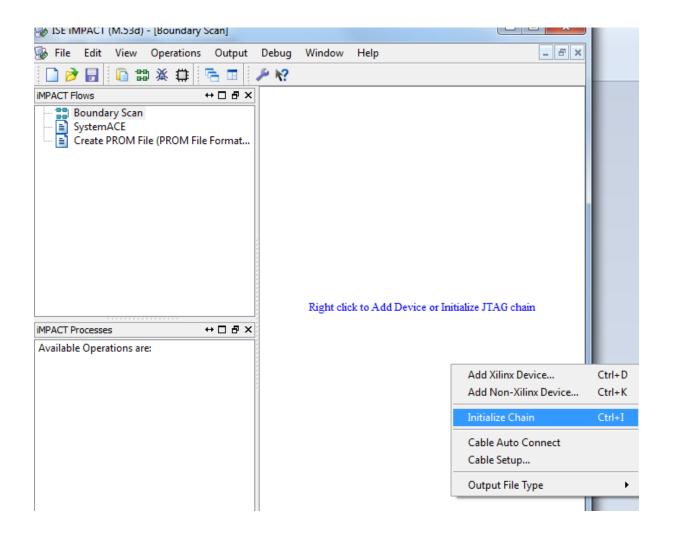
Xilinx ISE Design Suite 12.1
Accessories
퉬 ChipScope Pro
Documentation
🔒 EDK
퉬 ISE Design Tools
32-bit Project Navigator
54-bit Project Navigator
🍌 32-bit Tools
퉬 64-bit Tools
ar Constraints Editor
🏹 CORE Generator
🔯 FPGA Editor
impact
陷 Simulation Library Compilatio
🔞 Timing Analyzer
XPower Analyzer
4 Back
Search programs and files
🔊 ⋵ 🚞 🖸

Cancel all the initial prompts that come up.

Turn on the ML506. If this is your first time turning it on, Windows might automatically install some drivers (allow it to, a restart maybe required).

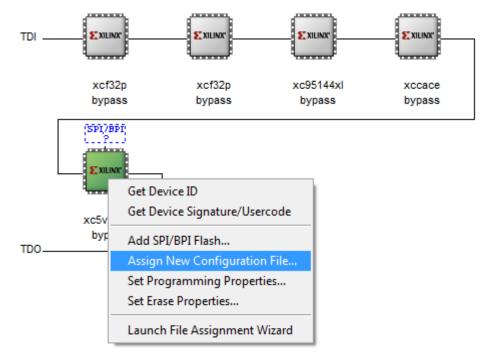
Double-click on 'Boundary Scan', then right-click on the empty plane and click 'Initialize Chain'. If you are getting errors about iMPACT unable to see your board, unplug the USB cable from your ML506 and replug it to initiate some windows auto driver install.

Cancel all the automatic prompts that come up.



If all is well, you should see the JTAG chain with the Virtex 5 FPGA as well as some PROMs, ACE controller and CPLDs.

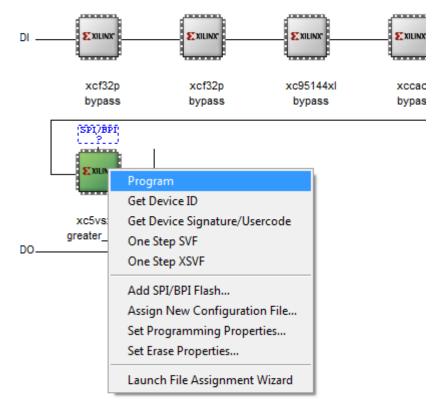
Right-click on the FPGA (xc5vsx50t) and click 'Assign New Configuration File'



then browse to your .bit file.

Name	Date modified	Туре	Size
퉬 _ngo	7/15/2011 10:16 AM	File folder	
퉬 _xmsgs	7/15/2011 10:19 AM	File folder	
퉬 ipcore_dir	7/12/2011 10:29 AM	File folder	
퉬 iseconfig	7/15/2011 10:15 AM	File folder	
퉬 xlnx_auto_0_xdb	7/15/2011 10:16 AM	File folder	
퉬 xst	7/15/2011 10:15 AM	File folder	
greater_2bit.bit	7/15/2011 10:19 AM	BIT File	2,444 KB





That's it! Your 2-bit greater-than circuit should now be operational. Input a is on switch 1 and 2 of the GPIO DIP switch and input b is on switch 3 and 4. Toggle the switches (the tip of a mechanical pencil would help if the switches are too small for your fingers) and if a is greater than b (in binary) then the GPIO led0 will light up, else it will stay dark.

. LEDO lights up when A is greater than B In this case A = '10' = 2 B = '00' = 0, so A is greater than B Inc A = '01' = 1 B = '10' = 2 So A is less than B A is less than B so LEDO stays dark

Hardware Co-Sim with a System Generator created Design Using a Black Box – 2-bit greater-than from above

Summary

This builds off of a tutorial I found here:

http://myfpgablog.blogspot.com/2009/12/sysgen-create-new-hwcosim-target-with.html

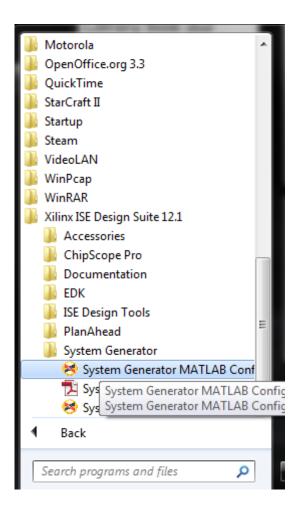
We will use the 2-bit greater-than circuit that was designed in the previous topic 'Programming the FPGA using a VHDL design...' Also the Xilinx blockset in Simulink will be used to define our circuit inputs and outputs. The logic will be fulfilled by using the 'Black Box' block containing our greater-than VHDL code. The bitsream will be created with System Generator and the design will be simulated via hardware co-simulation.

After finishing this tutorial you will be able to create a design using the Xilinx Blockset in Simulink and add a pre-made hdl design using a Black Box, then simulate the design on hardware within Simulink via hardware co-sim.

Creating the Design

First make sure that Xilinx System Generator is configured to use your version of MATLAB.

Run the System Generator MATLAB configurator as administrator (right-click \rightarrow run as administrator):



Select the version of MATLAB you will be using and click ok:

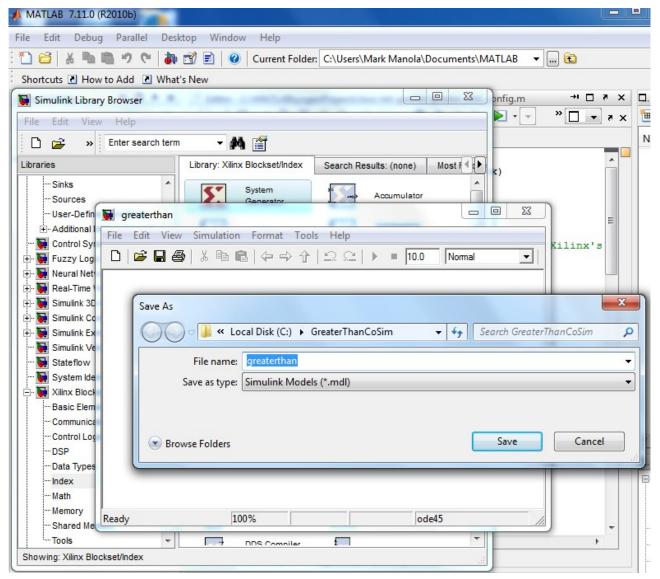
oose MATLAB for Sy	stem Generator 12.1	
ersion	Status	Location
7 📣 R2010a	🔀 Configured	C:\Program Files\MATLAB\R2010a
× -	<u> </u>	

Now open up MATLAB and then Simulink.

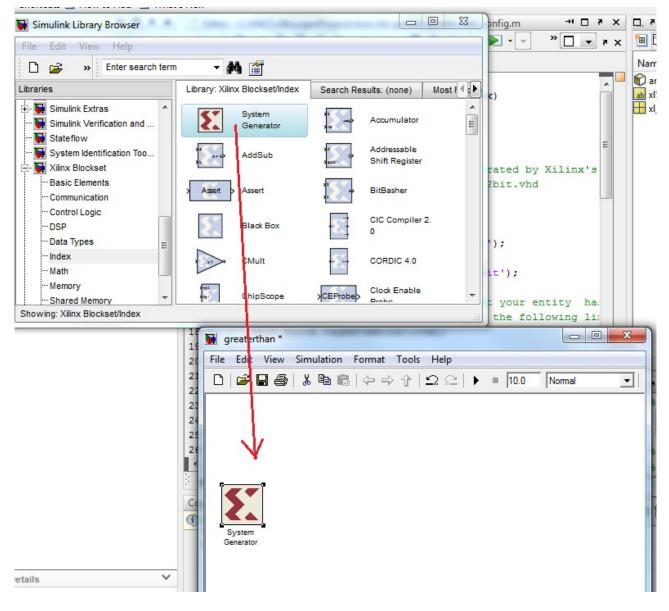
I flore a sector de la sector d	
🐌 Fraps	^
퉬 Games	
퉬 Guitar Pro 5	
퉬 iTunes	
퉬 Logitech	
퉬 Maintenance	
J MATLAB	
퉬 R2010a	
📣 Activate MATLAB R2010a	
📣 Deactivate MATLAB R2010a	
📣 MATLAB R2010a	
Microsoft Visual Studio 2010 Express	
🌗 ModelSim PE Stud Start MATLAB - The L	ang
퉬 Motorola	Ξ
퉬 OpenOffice.org 3.3	
퉬 QuickTime	
퉬 StarCraft II	
퉬 Startup	
퉬 Steam	
퉬 VideoLAN	Ŧ

A MATLAB 7.10.0 (R2010a)	
File Edit View Debug Paral	el Desktop Window Help
🔁 😂 👗 🖷 👘 🤊 🝽 🗋	🖡 🗊 📄 🕜 🛛 Current Folder: C:\MATLAB
Shortcuts 🖸 How to Add 💽 Wh	Simulink
Current Folder 🛛 😁 🗙	Workspace
🗢 🔿 🍌 « M 🕨 🔻 🔎 🌣 -	🛅 📷 🗃 骗 🧠 🛛 Stack: 🛛 Base 👻 🔯
Name	Name 🔺 Value
 AudioProcessing EE368ImageProcessing EE556Final EE556Midterm1 EM_Programs GUI_Programs Image Processing ImageProcessing RandomPrograms test1.asv piano_call.m piano.m test1.m EM_Programs.rar 	Command History New to MATLAB? Watch this Video, see D Warning: Userpath must be as fx >>

Once Simulink is open make a new model by clicking file \rightarrow New \rightarrow Model. Make a new folder on your local drive called 'GreaterThanCoSim' then save the model as 'greaterthan' into your newly created folder.



In the Simulink Library Browser scroll down to Xilinx Blockset and double-click it, then double click on the Index tab. You should see all the available Xilinx blocks to create your design. Click and drag the block 'System Generator' into your model editor.

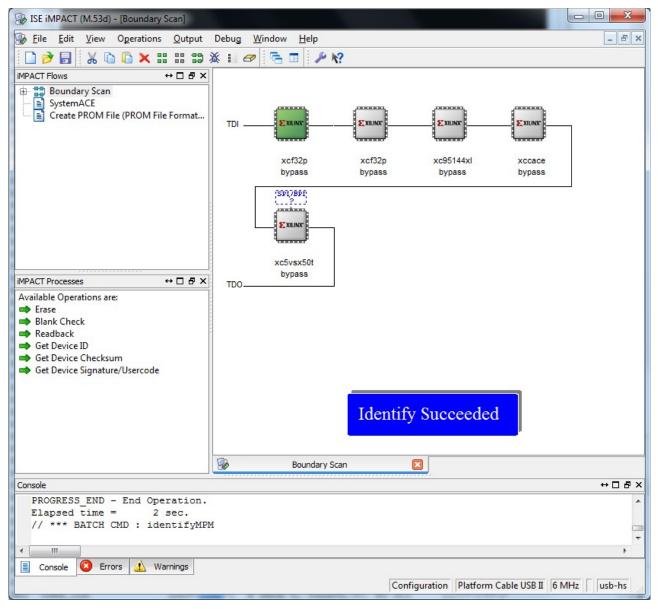


Now we will create a new compilation target so we co-simulate with the ML506. In your model editor window, double-click on the 'System Generator' block. Under Compilation click Hardware Co-Simulation \rightarrow New Compilation Target.

	Generator: greaterthan	
Compilati		
Part	HDL Netlist NGC Netlist Bitstream EDK Export Tool	
Clocking	Hardware Co-Simulation	ML402 ML506 ML506_GPIO_test ML506_greaterThan ML605 SP601 SP605 Spartan-3A DSP 1800A Starter Platform
	implementation : D(Spartan-3A DSP 3400A Development P
Clock En		XtremeDSP Development Kit
Provid	de clock enable clear pin	New Compilation Target
	ide with doubles : Accord	ting to Block Settings
Block	icon display: Default	-
Ger	nerate OK Apply	Cancel Help

Now the System Generator Board Description Builder should open up. Under Board Name type 'ML506_twobitGreater'. Set the system clock Frequency to 100MHz and the pin location to AH15 which is the USER_CLK pin.

To find the Boundary Scan Position turn on your board and open up iMPACT (see previous tutorial) after initializing the JTAG chain you will see this:



From here we see that the FPGA is in position 5 in the chain.

Leave the board on and click 'Detect' to get the IR Lengths.

On the Targetable Devices section, click Add \rightarrow virtex5 \rightarrow xc5vsx50t \rightarrow -1 \rightarrow ff1136 It should now look similar to this:

😝 System Gene	rator	Board Desc	ription Buil	der			X
Target Board Information							
Board Name ML506_twobitGreater							
System Clock							
Frequency (MHz) 100 Pin Location Ah15 Differential							
JTAG Options							
Boundary Sca	Boundary Scan Position 5 IR Lengths 16, 16, 8, 8, 10 Detect						Detect
Targetable Devices							
Family				Add >			
virtex5	xc5vsx50t -1				ff1136	[Delete
Non-Memory-N	Mappe	ed Ports					
Port Name		Direction		Wid	th		Add
							Edit
							Delete
Help	Loa	ad	Save Zip		Install		Exit

Now it is time to add our inputs and outputs. We will have two inputs each 2-bits wide and one output that is 1-bit wide.

Under the 'Non-Memory-Mapped Ports' section click Add..

Name the Port Name 'a' and select it as an input. Then for Pin LOC type: AG27 and click Add Pin. This will be the first bit of input a. Do the same for the second bit which is located on pin U25. After adding the second bit, click Save and Close.

Port Opti	Port Name	а		Input 🔘	Output
New Pin-	Pin LOC	PULI	LUP 🔲 PULL	DOWN	Add Pin
Pin List				2	_
Index	Pin LOC	PULLUP	PULLDO	FAST	Move Up
0	AG27 U25				Move Do
1				-	
1					Delete Pin
1					Delete Pin

Now we do the same for input b.

Port Opti	Port Name	b		Input 🔘 (Dutput
New Pin-	Pin LOC	PUL	LUP 🔲 PULL	DOWN	Add Pin
Index	Pin LOC	PULLUP	PULLDO	FAST	Move Up
0	AF26 AF25				Move Do
					Delete Pin
					Delete Pir

Click Save and Close, then click 'Add	' to add the output which will be sent to the	GPIO led 0
---------------------------------------	---	------------

	Port Name	agreatb) Input 🔘	Output
	Pin LOC	PULI	LUP 🖻 PULL	DOWN	Add Pin
Pin List	Pin LOC	PULLUP	PULLDO	FAST	Mauralla
)	H18	POLLOP			Move Up Move Do
					Delete Pin

Your screen should now look like this:

-	Information me ML506_twol	bitGreater			
System Clock					
Frequency (N		Pin Lo	cation A	h15 📃 🛙	Differential
JTAG Option	s				
Boundary S	can Position 5	IR Len	gths 16, 1	.6, 8, 8, 10	Detect
Targetable D	evices				
Family	Part	Speed		Package	Add >
virtex5	xc5vsx50t	-1		ff1136	Delete
Non-Memor	y-Mapped Ports				
Port Name	Directio	on	Widt	h	Add
a	in		2		Edit
b	in		2		Ealt
agreatb	out		1		Delete

Click Install to finish up the board description.

Once the installation is complete, a new model editor will open up with this model (you can now close the 'System Generator Board Description Window'):

Library: ml506_twobitgreater_lib *	
<u>File E</u> dit <u>V</u> iew F <u>o</u> rmat <u>H</u> elp	
□ ☞묘종 %┗偈 ⟨+ 수 ↑ 으으 ┡찒面	
nmm a> >agreatb	
nmm b>	
Ready 100%	Unlocked

In order to use this template we have to save it first. Click File \rightarrow Save as and save it in the 'GreaterThanCoSim' folder.

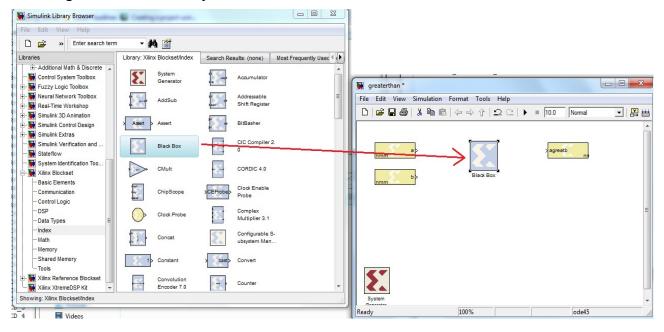
Now highlight the template and drag it into your 'greaterthan' model.

Library: ml506_twobitgreat	ter lib	_ 0 _ X
	- Help	
	💼 (수 수 수 으 으 📮 🗟 📷	
	>agreatb	
nmm b>		
	1	
	1	
	(reason	
Ready	100%	Unlocked
greaterthan *	Name I., Samer, M. H. analy W.	
	Format Tools Help	
0 🖻 🖬 🚭 👗 🖻 🌗	(≑ ⇒) 1 <u>∩</u> Ω ► = 10.0 Normal	
\checkmark		
a	>agreatb	
R		



Next we will add the logic to our circuit model. You can do this using blocks by going to the 'Simulink Library Browser' window and selecting the blocks you need. But since we already have our greater-than logic (in VHDL) we will use this to save some time.

In order to use a VHDL design in Simulink, we need the 'Black Box' block located under Xilinx Blockset \rightarrow Index in the Simulink Library Broswer.



So drag the Black Box into your model editor.

After dragging in the box, a window will automatically appear requesting the entity description for the black box. Locate the VHDL file for the greater-than circuit and click open:

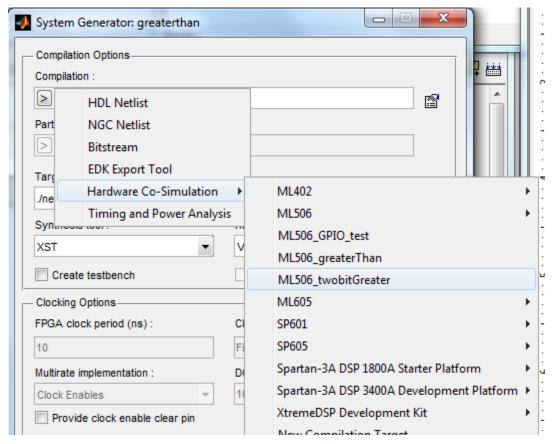
				-
Organize 🔻 New folder			III 🔹 🗖	(
🔆 Favorites	Name	Date modified	Туре	Size
🧮 Desktop	鷆 _ngo	7/15/2011 10:16 AM	File folder	
\rm Downloads	🎍 _xmsgs	7/15/2011 10:19 AM	File folder	
🖳 Recent Places	퉬 ipcore_dir	7/12/2011 10:29 AM	File folder	
	퉬 iseconfig	7/15/2011 10:15 AM	File folder	
詞 Libraries	inx_auto_0_xdb	7/15/2011 10:16 AM	File folder	
Documents	鷆 xst	7/15/2011 10:15 AM	File folder	
J Music	greater_2bit.vhd	7/12/2011 10:50 AM	VHD File	
Videos				
🚢 Local Disk (C:)				
👌 DVD/CD-RW Drive (I				
📬 Network	T			
		III		
•				

🖬 greaterthan *			
<u>File Edit View Simulation</u>	F <u>o</u> rmat <u>T</u> ools <u>H</u> elp		
0 🖙 🖬 🎒 🕺 🖻 🖻	⇔⇒∱ Ω⊆ ▶	I0.0 Normal	
a nmm b	Black Box	> agreatb nr	
System Ready	100%	ode45	•

Now your Black Box should have inputs and outputs as described by our VHDL file.

Wire up your inputs and outputs to the Black Box by clicking and holding down the left mouse button.

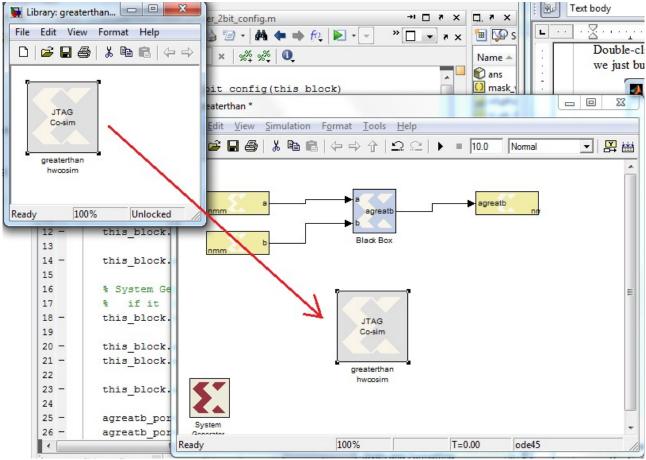
Double-click on the System Generator icon in your model editor and select the board description we just built under the Compilation tab.



If you don't see it, close the System Generator window and save your model file. Then double-click on the System Generator icon again.

After selecting 'ML506_twobitGreater' click Generate. This may take a while (5-10 mins).

Once completed, System Generator will show a Co-Sim block, drag this block into your model editor.



We can now simulate our design. In your model editor, click Simulation \rightarrow Configuration Parameters.

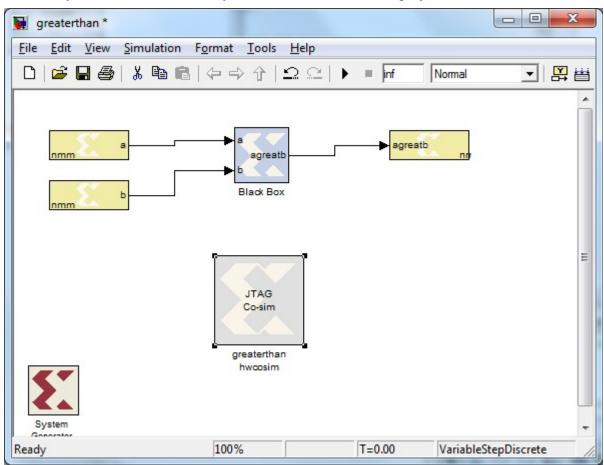
Set the stop time to: inf

Set Solver to : discrete

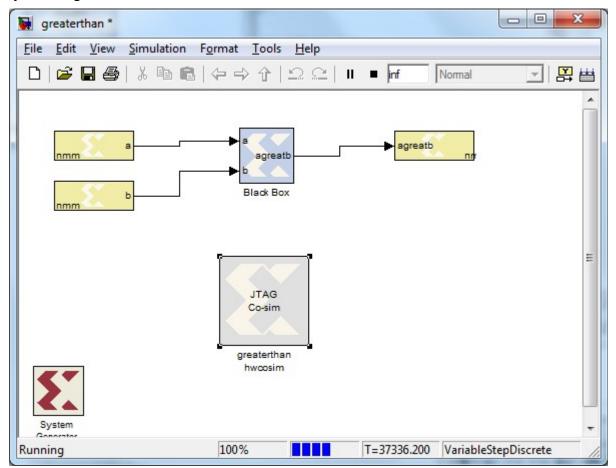
Select: Simulation time Solver Stat time: Data Import/Export Start time: Optimization Solver options Diagnostics Type: Variable-step Solver: discrete (no continuous states) Max step size: -Connectivity Tasking and sample time options -Compatibility Tasking mode for periodic sample times: -Model Referencing Saving -Stateflow Automatically handle rate transition for data transfer Higher priority value indicates higher task priority Zero-crossing options Zero-crossing control: Use local settings Algorithm: -Symbols Custom Code -Report Time tolerance: 10*128*eps Signal threshold: -Symbols Consecutive zero crossings: 1000 -Custom Code Debug Time tolerance: 10*128*eps Signal threshold: Debug Interface Mumber of consecutive zero crossings: 1000	Configuration Parameters: gre	eaterthan/Configuration (/	Active)	_		_Σ	×
Data Import/Export Start time: 0.0 Stop time: inf Optimization Solver options Plagnostics Type: Variable-step Solver options Max step size: auto • Oata Validity Max step size: auto • Connectivity Tasking and sample time options • Compatibility Tasking mode for periodic sample times: Auto Model Referencing • Tasking mode for periodic sample times: Auto Simulation Target • • Higher priority value indicates higher task priority Model Referencing Zero-crossing options Zero-crossing options Zero-crossing options Simulation Target • 10*128*eps Signal threshold: auto -Report Comments • 10*128*eps Signal threshold: auto -Symbols • 1000 • •	Select:	Simulation time					-
Data Import/Export Optimization Solver options Diagnostics Type: Variable-step Solver: discrete (no continuous states) Tasking and sample time options Connectivity -Compatibility Model Referencing Saving Automatically handle rate transition for data transfer Hardware Implementati Model Referencing Simulation Target Symbols Custom Code Pebug Interface Interface		Start time: 0.0		Stop time: inf			
Diagnostics Type: Variable-step Solver: discrete (no continuous states) - Data Validity Max step size: auto Max step size: auto - Type Conversion - Connectivity - Connectivity - Connectivity - Connectivity - Connectivity - Connectivity - Compatibility Tasking and sample time options - Saving - Stateflow - Hardware Implementati - Higher priority value indicates higher task priority - Model Referencing - Simulation Target - Symbols - Custom Code - Debug - Interface - Iff - Time - Value - Value - Value - Value - Solue - Value - Value							
-Sample Time Type: Variable-step Solver: discrete (no continuous states) -Type Conversion -Connectivity -Compatibility Tasking and sample time options -Saving -Sateflow -Stateflow -Automatically handle rate transition for data transfer -Stateflow -Higher priority value indicates higher task priority Model Referencing -Solver: -Stateflow -Automatically handle rate transition for data transfer -Simulation Target -Symbols -Custom Code -Debug -Interface -Ime to consecutive zero crossings: Image: Number of consecutive zero crossings: 1000		Solver options					
Output Number Data Validity -Data Validity Max step size: auto -Type Conversion Connectivity -Compatibility Tasking and sample time options -Model Referencing Saving -Stateflow Automatically handle rate transition for data transfer -Hardware Implementati Higher priority value indicates higher task priority -Model Referencing Zero-crossing options -Symbols Custom Code -Comments Symbols -Custom Code Number of consecutive zero crossings: -Symbols 1000	1 I I I	Type: Varia	ble-sten	Solver: discrete	(no continuous states)	-	
 Type Conversion Connectivity Compatibility Model Referencing Saving Saving Saving Saving Automatically handle rate transition for data transfer Higher priority value indicates higher task priority Zero-crossing options Zero-crossing control: Use local settings Algorithm: Nonadaptive Time tolerance: 10*128*eps Signal threshold: auto Number of consecutive zero crossings: 1000 			ine step				
Connectivity Compatibility Model Referencing Saving Saving Sateflow Hardware Implementati Model Referencing Simulation Target Use local settings Algorithm: Nonadaptive Time tolerance: 10*128*eps Signal threshold: auto Number of consecutive zero crossings: 1000	· · · · · · · · · · · · · · · · · · ·	Max step size: auto					
 Compatibility Model Referencing Saving Stateflow Hardware Implementati Model Referencing Simulation Target Symbols Custom Code Report Comments Symbols Custom Code Debug Interface 							
 Saving Stateflow Hardware Implementati Model Referencing Simulation Target Symbols Custom Code Real-Time Workshop Report Comments Symbols Custom Code Time tolerance: 10*128*eps Number of consecutive zero crossings: Number of consecutive zero crossings: 	· · · · · · · · · · · · · · · · · · ·	Tasking and sample t	ime options				Ξ
 Stateflow Hardware Implementati Model Referencing Simulation Target Symbols Custom Code Real-Time Workshop Report Comments Symbols Custom Code Time tolerance: 10*128*eps Number of consecutive zero crossings: Number of consecutive zero crossings: 	Model Referencing	Tasking mode for per	iodic sample times:	Auto		-	
Stateflow Hardware Implementati Model Referencing Simulation Target Symbols Custom Code Real-Time Workshop Report Comments Symbols Custom Code Debug Interface Im	Saving	Automatically har	dle rate transition for data transfer				
Model Referencing Simulation Target Symbols Custom Code Real-Time Workshop Report Comments Symbols Custom Code Debug Interface							
Simulation Target Zero-crossing options Symbols Zero-crossing control: Use local settings Algorithm: Nonadaptive Custom Code Time tolerance: 10*128*eps Signal threshold: auto Report Number of consecutive zero crossings: 1000 Costom Code Debug Interface	· ·	Higher priority val	ue indicates higher task priority				
Symbols Custom Code Custom Code Custom Code Comments Symbols Custom Code Debug Interface		Zero-crossing options	5				
Custom Code Real-Time Workshop Report Comments Symbols Custom Code Debug Interface Time tolerance: 10*128*eps Signal threshold: auto 1000 1000 1000 Time tolerance: 10*128*eps Signal threshold: auto 1000 Time tolerance: 10*128*eps Signal threshold: auto Time tolerance: 10*128*eps Time tolerance: 10*128*eps Signal threshold: auto Time tolerance: 10*128*eps Time toleran	1 i . "	Zoro-crossing control		Algorithm	Nonadaptivo		
Real-Time Workshop Report Comments Symbols Custom Code Debug Interface		-	Ū Ū				
Comments Symbols Custom Code Debug Interface	⊡-Real-Time Workshop	Time tolerance:	10*128*eps	Signal threshold:	auto		
Symbols Custom Code Debug Interface	Report	Number of consecutiv	e zero crossings:		1000		
-Custom Code -Debug -Interface	Comments						
Debug Interface							
Interface							
	Interrace						
							-
	•					+	
				<u>о</u> к	<u>C</u> ancel <u>H</u> elp <u>A</u>	<u>A</u> pply	

Click ok.

Make sure your board is on then in your model editor click the play button.



Simulink will now program your FPGA (this may take a while). Wait until the model editor screen says running..



Then you can toggle the GPIO switches to test the greater-than circuit, a picture is shown below:



Illustration 2: a = '10' = 2; b = '01' = 1; *thus a* > *b*

You can further customize this by adding more blocks to manipulate the signal. Here are some more complex design tutorials using a MAC FIR from Xilinx:

http://www.xilinx.com/products/boards/ml506/ml506_12.1/dsp.htm

Xilinx also provides a set of System Generator tutorials located in the installation directory. Mine is in

C:\\Xilinx\12.1\ISE_DS\ISE\sysgen\examples

Using Xilinx XPS and SDK to implement serial communication using the RS232 cable and a terminal program

Summary

This tutorial will show you how to use Xilinx XPS base system builder to create a base system package that connects the FPGA to its board peripherals such as the UART serial interface, GPIO LEDs, etc. After building the base system we will export it to Xilinx XPS where we can write the software to test the UART interface.

Note: I have been experiencing some crashes with the Xilinx SDK version 12.1 in Windows 7. Version 13.1 and later is supposed to support Windows 7, so for this tutorial I will use Xilinx EDK version 13.2.

Xilinx XPS

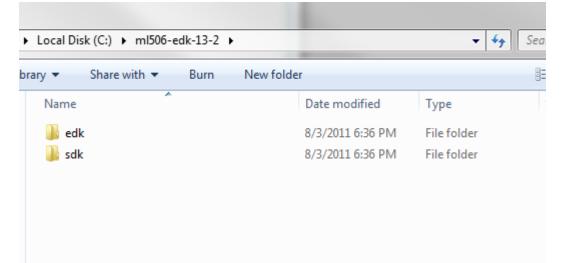
First we will use XPS and the system builder wizard to add peripherals to our FPGA. This is done by creating a soft microprocessor to handle peripheral interfacing and communicate with the FPGA.

First we will create a file system to organize our project files. Create a folder (I put mine in the local drive) and name it (make sure there are no spaces):

Organize 🔻 🛛 🔭 Open	Include in library + Share with + Bur	n New folder		•	(
🔆 Favorites	Name	Date modified	Туре	Size	
E Desktop	c_cpp_programs	6/2/2011 8:18 PM	File folder		
Downloads	🔒 code	4/8/2011 2:53 PM	File folder		
Recent Places	퉬 dev	7/4/2011 9:21 AM	File folder		
Recorded TV	📕 eclipse	7/30/2011 9:53 AM	File folder		
	\mu Fraps	8/3/2011 6:18 PM	File folder		
🔚 Libraries	MATLAB	7/14/2011 8:53 AM	File folder		
Documents	ModelSimTestbenches	6/30/2011 7:10 PM	File folder		
J Music	퉬 Modeltech_pe_edu_10.0a	6/20/2011 2:55 PM	File folder		
Pictures	🌗 PerfLogs	7/13/2009 8:20 PM	File folder		
Videos	퉬 Photoshop	8/3/2011 6:10 PM	11 6:10 PM File folder		
	🌗 Program Files	8/3/2011 6:30 PM	File folder		
🝓 Homegroup	🌗 Program Files (x86)	7/30/2011 1:05 PM	File folder		
	🌗 Simulink_Programs	5/21/2011 10:44 PM	File folder		
🖳 Computer	퉬 starcraftInstall	5/29/2011 10:26 AM	File folder		
🚢 Local Disk (C:)	🍺 testbench	7/13/2011 1:54 PM	File folder		
👝 Storage (D:)	Users	7/12/2011 12:50 PM	File folder		
	퉬 Windows	8/1/2011 7:31 PM	File folder		
📬 Network	퉬 WTablet	8/2/2011 3:13 PM	File folder		
	퉬 Xilinx	8/1/2011 4:42 PM	File folder		
	JilinxML506	8/3/2011 3:44 PM	File folder		
	xilinxProjects	7/11/2011 10:09 AM	File folder		
	📕 ml506-edk-13-2	8/3/2011 6:34 PM	File folder		

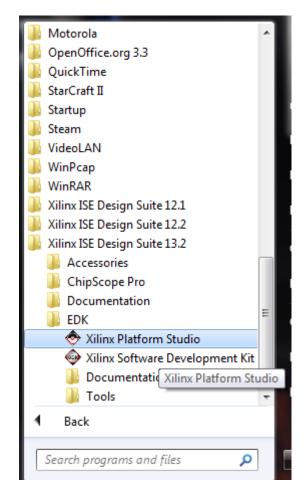
ml506-edk-13-2

Now within this folder make two empty folders name: edk, sdk



We will store the board configuration files in the edk folder and the software files in the sdk folder.

Open up Xilinx XPS (if you are using Windows 7, use Version 13.1 or later)



Choose Base System Builder wizard:

BSB	<u>B</u> ase System Builder wizard (recommended)
Č,	Blank XPS project
	Open a recent project
Brows	e for More Projects
_	DK examples (projects) on the web <u>here</u>

In the Project File section point it to the edk folder we created earlier and for the interconnect type choose PLB. For newer models of FPGA choose the AXI.

Click OK.

New Project	t
<u>P</u> roject File	C:\ml506-edk-13-2\edk\system.xmp Browse
Select an In	iterconnect Type
© AXI	I System
fut	I is an interface standard recently adopted by Xilinx as the standard interface used for all current and ture versions of Xilinx IP and tool flows. Details on AXI can be found in the AXI Reference Guide on nx.com.
PLB	3 System
Vir mi <u>c</u>	B is the legacy bus standard used by Xilinx that supports current FPGA families, including Spartan6 and tex6. PLB IP will not support newer FPGA families, so is not recommend for new designs that may grate to future FPGA families. Details on PLB can be found in the PLBv46 Interface Simplifications cument on xilinx.com.
Select Existi	ing .bsb Settings File(saved from previous session)
	Browse
Set Project	Peripheral Repository Search Path
Set Project	Peripheral Repository Search Path Browse

Choose new design and click Next.

Welcome	Board	System	Processor	Peripheral	Cache	Summa
/elcome to the Base Syst						
his tool leads you through the	steps necessary for	r creating an embedde	ed system.			
elect One of the Following:						
I would like to create a ne						
I would like to load an exis	sting .bsb settings fil	le (saved from a previ	ious session)			
						Browse
More Info				< Back	Next >	Cancel

Choose the development board, in our case it is a Virtex 5 ML506 Evaluation Platform and click next.

🐟 Base System Bui	lder					? X
Welcom	e Board	System	Processor	Peripheral	Cache	Summary
Board Selection Select a target devel	opment board.					
Board						
I would like to cro	eate a system for the follow	wing development board				
Board Vendor	Xilinx					•
Board Name	Virtex 5 ML506 Evaluation	Platform				-
Board Revision	1					•
I would like to create the second	eate a system for a custom	board				
Board Information						
Architecture	Device		Package	Spe	ed Grade	
virtex5		50t		-1		T
Use Stepping						-
Reset Polarity Activ	e Low					-
Related Information						
Vendor's Website						
Vendor's Contact Info	ormation					
Third Party Board De	finition Files Download Web	site				
board includes Tri-Mo	Itended to showcase and d de Ethernet MAC/PHY, 256 ctors, System ACE CF cont	MB DDR2 SDRAM SODIM	IM memory, 1MB ZBT SR			
More Info				< <u>B</u> ack	Next >	Cancel

The Base System Builder will create a soft processor for interfacing with the peripherals, choose Single-Processor System and click Next.

Welcome	Board	System	Processor	Peripheral	Cache	Summar
ystem Configuration onfigure your system.						
	Sin <mark>gle-</mark> Processor Syst	em	1.1	Dual-Process	or System	
Select this option to creat will let you configure the configuration parameters	processor, the periph	gle processor. This Wizard eral set and some major	will let you sele	on to create a design wi ct the types of process peripherals shared by t	ors, peripherals unique	
Processor 1			Process		or 1 Peripherals	
	Processor 1 Pe RS232 GP			Shared Mailbox	Peripherals Mutex	
			Process		or 2 Peripherals	

Base System Builder						? <mark>x</mark>
Welcome	Board	System	Processor	Peripheral	Cache	Summary
Processor Configuration	1					
Configure the processor(s).						
Reference Clock Frequency	100.00					→ MHz
Processor 1 Configuration						
Processor Type	MicroBlaze					•
System Clock Frequency	125.00					▼ MHz
Local Memory	8 KB					-
Debug Interface	On-Chip HW Debug I	Module				T
Enable Floating Point	Unit					
More Info				< <u>B</u> ack	Next >	Cancel

Choose these default values for the processor specs and click Next.

In the Peripheral window, leave it default. As you can see there are many peripherals connected to the MicroBlaze soft processor. We only need to create a base system once and we can reuse it for different projects so we might as well connect all the peripherals in case we need to use one for another project. Click Next.

🔶 Base System Builder		_	_			? ×
Welcome	Board	System	Processor	Peripheral	Cache	Summary
Peripheral Configuration To add a peripheral, drag it from Available Peripherals Peripheral Names	the "Available Per	ripherals" to the pro	cessor peripheral list. To Processor 1 (MicroBl		click on the periphe	eral.
FLASH Hard_Ethernet_MAC Internal Peripherals Imb_bram_if_cntlr xps_bram_if_cntlr xps_timebase_wdt xps_timer		Add > < Remove	DDR2_SDRAM Core DIP_Switches_8Bit Core: xps_gpit Ethernet_MAC Core: xps_ethi IIC_EEPROM Core: xps_git LEDs_8Bit Core: xps_gpit LEDs_Positions Core: xps_gpit PCIe_Bridge Core: plbv46_ Push_Buttons_5Bit Core: xps_gpit RS232_Uart_1	o ernetlite o p pcie t	mpmc	
			RS232_Uart_2	Flash ace um_if_cntIr		
More Info				< <u>B</u> ack	Next >	Cancel

Base System Builder		-				? ×
Welcome	Board	System	Processor	Peripheral	Cache	Summar
Cache Configuration Gelect cache size and cache m	emory for processo	r(s).				
-Processor 1 (MicroBlaze) Ca						
In MicroBlaze, caches are o	ptional and configur	able. Caches are imple	emented using FPGA LU		ck RAMs for large size	ed caches.
Instruction Cache Size 2	KB		Data Cache S	ze 2 KB		T
Instruction Cache Memory	,		Data Cache M	emory		
SRAM			SRAM			
DDR2_SDRAM			O DDR2_S	DRAM		
More Info				< <u>B</u> ack	Next >	Cancel

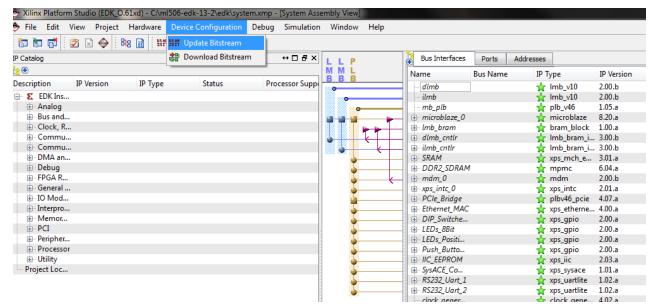
Click Next on the Cache window as we won't use any.

The next page is the summary of our design, review it if you like and click Finish. (Make sure to check the Save Base System Builder Settings File box).

Welcome	Board	System	Processor	Peripheral	Cache	Summa
Summary						
elow is the summary of th	e system you are creating.					
stem Summary						
Core Name	Instance Name	Base Address	High Address			
 Processor 1 mpmc xps_gpio xps_ethernetlite xps_iic xps_gpio xps_gpio xps_gpio plbv46_pcie xps_gpio xps_uartlite xps_uartlite 	microblaze_0 DDR2_SDRAM DIP_Switches_8Bit Ethernet_MAC IIC_EEPROM LEDs_8Bit LEDs_Positions PCIe_Bridge Push_Buttons_5Bit RS232_Uart_1 RS232_Uart_2	0x9000000 0x81460000 0x8100000 0x8160000 0x81440000 0x81420000 0x85C00000 0x81400000 0x84020000 0x8400000	0x9FFFFFF 0x8146FFFF 0x8100FFFF 0x8160FFFF 0x8144FFFF 0x8142FFFF 0x85C0FFFF 0x8140FFFF 0x8402FFFF 0x8400FFFF			
···· C:\ml506-edk-13 ···· C:\ml506-edk-13 ···· C:\ml506-edk-13 ···· C:\ml506-edk-13	SRAM SysACE_CompactFlash Ir dlmb_cntlr -2\edk\system.xmp -2\edk\system.mhs -2\edk\data\system.ucf -2\edk\etc\fast_runtime.cc -2\edk\etc\download.cm -2\edk\etc\bitgen.ut	0x0000000	0x8AFFFFF 0x8360FFFF 0x00001FFF			
	er (.bsb) Settings File					

The base system is now finished but before we can write some software for it we have to update the bitstream and export the board configurations to Xilinx SDK.

Under Device Configuration click Update Bitstream, this process takes a long time (around half an hour).



The console will say done when the bitstream update is complete:

-				111	•	Legend Master Slave Master/Sla Production License (paid
4		-			P	ASuperseded ODiscontinu
۲	Project	🔶 IP Catalog				🔶 Start I
0	nsole					
Con	ISOIE					
	Memory	Initializat	ion	completed successfully.		
	Done!					
-						
		A	<u> </u>	-		
	Console	🚹 Warnings	U	Errors		

📀 Xilinx Platform Stud	o (EDK_O.61xd) - C:\ml506-edk-13-2\edk\system.xmp	o - [System Assembly View]
🔶 File Edit View	Project Hardware Device Configuration Debu	g Simulation Window Help
🖥 🕅 🗗 🖉 🛛	🛃 Project Options	
IP Catalog	Design Rule Check Ctrl+Shift+D	↔□♂× LLP
1 <mark>2:</mark> 🖲	🔁 Select Elf File	M M L Nam
Description IP Ve	Export Hardware Design to SDK	cessor Suppl
EDK Ins		
⊕ Bus and	Archive Project	
⊕ Clock, R	□ Generate Block Diagram Image	
E Commu	Generate and View Design Report	• • • • • • •
⊕ Commu ⊕ DMA an		
	📡 View Design Summary	· ⊕ · S

To export this project to the SDK, click Project \rightarrow Export Hardware Design to SDK..

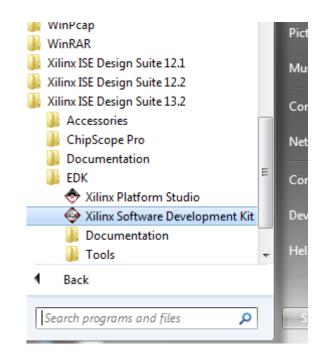
Check the box to include the bitstream file we just generated and the BMM file. Then click Export Only: (This will created some files located in C:\ml506-edk-13-2\edk\SDK\SDK_Export\hw)

Export to SDK / Launch SDK					
This dialog allows you to export hardware platform information to be used in SDK.					
Include bitstream and BMM file					
(XPS will regenerate bitstream if necessary, and it may take some time to finish.)					
Directory location for hardware description files					
SDK\SDK_Export					
Export Only Export & Launch SDK Cancel Help					

This will enable the Xilinx SDK to use the hardware configuration we just described and create software to utilize them.

Xilinx SDK

Close Xilinx XPS and open Xilinx SDK:



Point your workspace to the sdk folder we created at the beginning of this tutorial and click OK:

Workspace	Launcher		X
Select a w	orkspace		
	ores your projects in a folder called a workspace. orkspace folder to use for this session.		
<u>W</u> orkspace:	C:\ml506-edk-13-2\sdk		Browse
🕅 <u>U</u> se this a	s the default and do not ask again	ОК	Cancel

🛞 C	/C++ - Xilinx	SDK				-	P10		and the second second
File	Edit Sour	ce Refactor	Navigate	Search	Run	Project	Xilinx Tools	Window	Help
1	New		Alt	+Shift+N	ها ۱	Xilinx C	Project]
	Open File				4	Xilinx C	++ Project		
r	Close Close All		Ctrl+	Ctrl+W ⊦Shift+W		Xilinx Board Support Package			cation
	Save			Ctrl+S	63	 Source Folder Folder Source File Header File File from Template Class Other 			
R.	Save As Save All Revert		Ctrl	l+Shift+S	C				
	Move Rename			F2					
69	Refresh Convert Line	e Delimiters T	0	F5	, 📑		Ctrl+N		
.e.	Print			Ctrl+P			cumentation		
ð	Switch Work Restart	space			•				<u>Getting Started w</u> <u>EDK Concepts, T</u> <u>Migrating from old</u>
2	Import							 Frequently asked 	
4	Export							ĸn	own Issues
	Properties Exit		4	Alt+Enter					Known issues in Xilinx Answer Ree

Click File \rightarrow New \rightarrow Xilinx C Project:

The Xilinx SDK will prompt you to specify a hardware platform. This is the file we exported in XPS and it contains all the hardware info of our board. Click Specify:

🚳 No H	Hardware Platforms in the Workspace
2	SDK requires a hardware platform specification to support application development. The hardware platform specification is exported by either Project Navigator or Xilinx Platform Studio. Currently, no hardware platforms have been brought into the workspace. Select 'Specify' to specify one now, or 'Cancel' to cancel the wizard. Specify Cancel

Name the project UART_Test and point the hardware specification entry to the system.xml file that we exported. The bitstream and BMM entries should automatically fill after setting the system.xml entry: Click Finish

🐵 New Hardwa	re Project	
New Hardwa Create a new I	are Project Hardware Project.	E.
<u>P</u> roject name: ▽ Use defau		
Location: C:	mI506-edk-13-2\sdk\UART_Test	B <u>r</u> owse
This file usu The specific C:\ml506-e	path to the hardware specification file exported from Project Navigator ally resides in SDK/SDK_Export/hw folder relative to the XPS project loca ation file and associated bitstream content will be copied into the work dk-13-2\edk\SDK\SDK_Export\hw\system.xml and BMM Files	ation.
	you'd like to associate a bitstream different than the one that was expo ify that below.	orted from XPS,
Bitstream:	C:\ml506-edk-13-2\edk\SDK\SDK_Export\hw\system.bit	Browse
BMM File:	C:\ml506-edk-13-2\edk\SDK\SDK_Export\hw\system_bd.bmm	Browse
?	<u> </u>	Cancel

In the next window, name the project hello_uart and select the Empty Application template, then click Next:

😡 New Project		
New Xilinx C Project Create a managed make application project	. Choose from one of the sample applications.	G
Project name: hello_uart		
Use <u>default</u> location		
Location: C:\ml506-edk-13-2\sdk\hello_ua	art	Browse
Choose file system: default 👻		
Hardware Platform: UART_Test		
Processor: microblaze_0		
Dhrystone Empty Application Hello World IwIP Echo Server Memory Tests Peripheral Tests SREC Bootloader Xilkernel POSIX Threads Demo	Description A blank C project.	A T
?	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish	Cancel

Then name the board support package hello_uart_bsp and click Finish:

New Project						
New Xilinx C Pr Create a managed	roject d make application project. Choose from one of the sample applications.	5				
Oreate a new Board Support Package project						
The template	provided by application 'Empty Application' will be used to configure the project.					
Project name:	: hello_uart_bsp					
👿 Use <u>d</u> efau	It location					
Location: C:	\ml506-edk-13-2\sdk\hello_uart_bsp	B <u>r</u> owse				
Cho	oose file system: default 💌					
Available Board	ng Board Support Packages d Support Packages found					
?	< <u>B</u> ack <u>N</u> ext > <u>Finish</u>	Cancel				

Xilinx SDK will create a template for your new C project.

Expand the 'src' folder under the 'hello_uart' project folder, then right-click on 'src' and click New \rightarrow Source File.

Name the source file hello uart.c and click Finish:

New Source	File		
Source File Create a new s	ource file.		C
Source fol <u>d</u> er: Source fil <u>e</u> :	hello_uart/src hello_uart.c		<u>B</u> rowse
<u>T</u> emplate:	Default C source template	•	Configure
?		Finish	Cancel

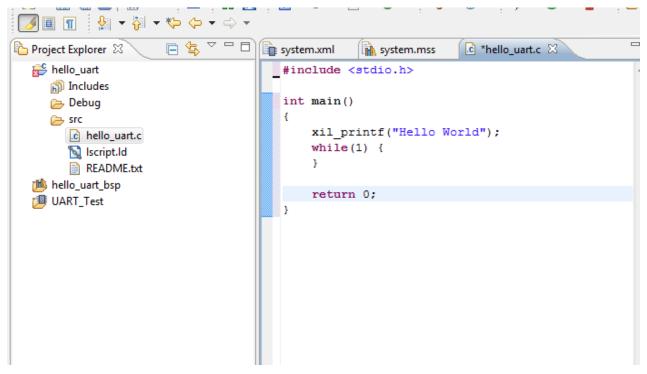
A Xilinx text editor should come up (if it doesn't, double click hello_uart.c in the Project Explorer windows), delete all the auto generated comments and type in this code:

```
#include <stdio.h>
```

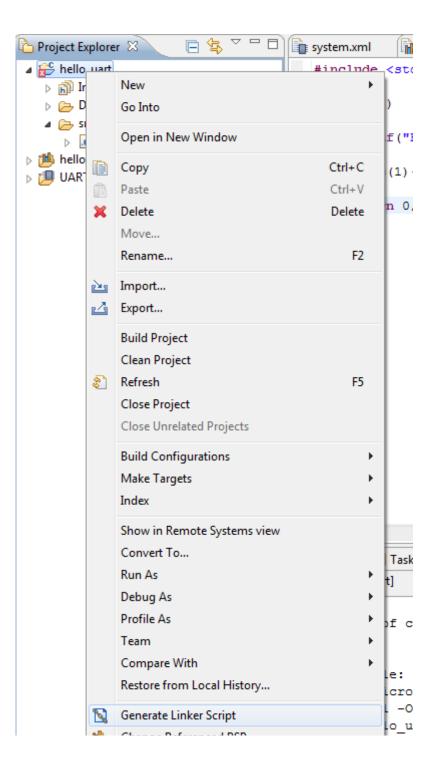
```
int main()
{
    xil_printf("Hello World");
    while(1) {
    }
    return 0;
}
```

Note: use xil_printf() when you can, printf () takes up too much memory space.

Click save and Xilinx SDK should automatically build and compile your code and create a file called hello_uart.elf which you can use to program the FPGA.



Now right-click on the hello_uart project and click Generate Linker Script



Leave all the default settings and click Generate

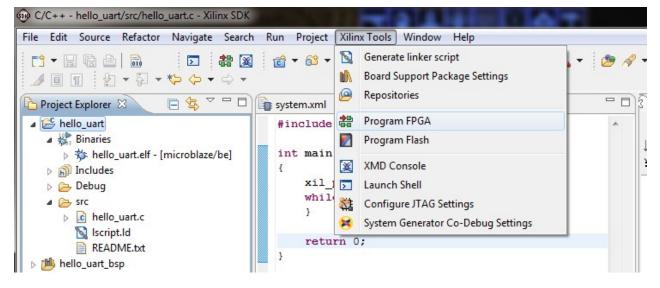
Note that if your .elf file is too big, you might have to increase the Heap and Stack size to accommodate your code:

nerate linker script						2
ontrol your application's memory m	ap.					
				Basic Advanced		
roject: hello_uart				Place Code Sections in:	ilmb_cntlr_dlmb_cntlr	
utput Script: C:\ml506-edk-13-2\sdk\hello_uart\s	rc\lscrint ld		Browse	Place Data Sections in:	ilmb_cntlr_dlmb_cntlr	
			browse	Place Heap and Stack in:	ilmb_cntlr_dlmb_cntlr	
lodify project build settings as follov Set generated script on all project bu]	1000 mm 1		
set generated script on all project bu	lind configurations		•	Heap Size:	1 KB	
				Stack Size:	1 KB	
Memory	Base Address	Size				
ilmb_cntlr_dlmb_cntlr	0x00000000	8 KB				
SRAM_MEMO_BASEADDR	0x8AF00000 0x90000000	1 MB 256				
DDR2_SDRAM_MPMC_BASEADDR	0x9000000	200				
Fixed Section Assignments						

This .elf file, along with some system configuration bitstreams, is what you need to program the FPGA and run your software.

Turn on the ML506 and open up a serial terminal program such as HyperTerminal, TeraTerm, Putty etc.. Here I use TeraTerm, and set it up for serial communication on COM1 with 9600 baude:

Click on Xilinx Tools \rightarrow Program FPGA:

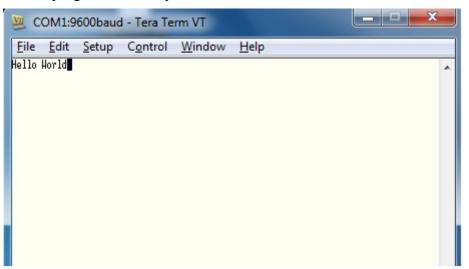


Point the Bitstream and BMM File entries to the files exported by Xilin XPS and under the microblaze_0 processor entry choose the hello_uart.elf file:

Program I	FPGA X	
Program I	FPGA -	հ
Specify the	bitstream and the ELF files that reside in BRAM memory	ī
Hardware S Bitstream: BMM File:	pecification: C:\ml506-edk-13-2\sdk\UART_Test\system.xml C:\ml506-edk-13-2\edk\SDK\SDK_Export\hw\system.bit C:\ml506-edk-13-2\edk\SDK\SDK_Export\hw\system_bd.bmm Browse	
Processor	ELF File to Initialize in Block RAM	
microblaze	_0 C:\ml506-edk-13-2\sdk\hello_uart\Debug\hello_uart ▼	
?	Program Cancel	

Click Program and Xilinx SDK will combine the .bit, .bmm, .elf, files into a single bitstream called 'download.bit' located in C:\ml506-edk-13-2\sdk\UART_Test.

I suggest you rename this file to something more meaningful like hellouart.bit.



The FPGA is now programmed and you can see the results in the terminal window:

Play sound with a sine wave using the AC97 codec

Summary

This tutorial makes use of the AC97 codec, LCD, UART (for text display) to play some sine waves through the line-in port of the AC97 interface. User instructions will be displayed to a COM serial window such as HyperTerminal or TeraTerm. Also some simple instruction will be displayed to the user via the on-board LCD screen.

To avoid reinventing the wheel, we will use some Xilinx pre-made board configuration files and software.

Xilinx XPS

In order to use the on-board AC97 codec we would have to read the datasheet for the codec available here:

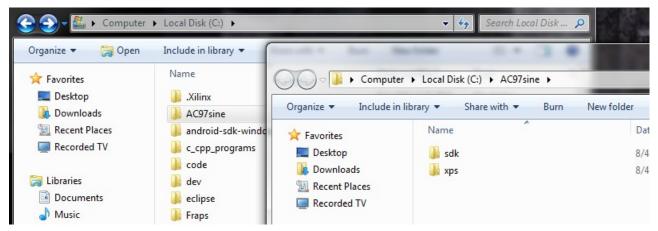
http://www.xilinx.com/products/boards/ml505/datasheets/87560554AD1981B_c.pdf

Then we would have to use our MicroBlaze softprocessor to setup AC97 according to its datasheet.

But Xilinx has already done this for use so we can just use theirs and tailor it to our needs.

First lets setup our file system to organize our project.

Create a folder on the local drive called: AC97sine then within this folder create two more named sdk and xps.



The Xilinx XPS files will be stored in the xps folder and the Xilinx SDK workspace will point to the sdk folder.

First we need to download the pre-made Xilinx files.

Get them here:

http://www.xilinx.com/products/boards/ml506/ml506_12.1/bsb.htm Choose the ML506 EDK Standard IP Design with Pcores Addition .zip file

Organize 🔻 🛛 Include in		folder		0
🔆 Favorites	Name	Date modified	Туре	Siz
🧮 Desktop	🔰 bootloops	8/4/2011 5:25 PM	File folder	
属 Downloads	🔰 data	8/4/2011 5:25 PM	File folder	
🖳 Recent Places	🔰 etc	8/4/2011 5:25 PM	File folder	
Recorded TV	🍌 implementation	8/4/2011 5:25 PM	File folder	
	inicroblaze_0	8/4/2011 5:25 PM	File folder	
🔰 Libraries	🌗 pcores	8/4/2011 5:25 PM	File folder	
Documents	🎉 ready_for_download	8/4/2011 5:25 PM	File folder	
J Music	鷆 sw	8/4/2011 5:25 PM	File folder	
E Pictures	TestApp_Memory_microblaze_0	8/4/2011 5:25 PM	File folder	
Videos	ml506_bsb_system.bsb	5/3/2010 10:59 AM	BSB File	
	ml506_bsb_system.mhs	5/12/2010 12:35 PM	MHS File	
🖏 Homegroup	ml506_bsb_system.mss	5/12/2010 12:35 PM	MSS File	
	🗇 ml506_bsb_system	5/12/2010 12:43 PM	Xilinx Platform Stu	
📮 Computer				
🏭 Local Disk (C:)				
👝 Storage (D:)				
📮 Network				
	•	m		

Extract the files somewhere like the desktop or a temp directory:

Organize 🔻 🛛 🔭 Open	Burn New folder		•	2
🔶 Favorites	Name	Date modified	Туре	Si
🧮 Desktop	🌗 bootloops	8/4/2011 5:26 PM	File folder	
\rm Downloads	🔒 data	8/4/2011 5:26 PM	File folder	
🕮 Recent Places	🔒 etc	8/4/2011 5:26 PM	File folder	
Recorded TV	implementation	8/4/2011 5:26 PM	File folder	
	imicroblaze_0	8/4/2011 5:26 PM	File folder	
🔰 Libraries	pcores	8/4/2011 5:26 PM	File folder	
Documents	ready_for_download	8/4/2011 5:26 PM	File folder	
J Music	🔋 sw	8/4/2011 5:26 PM	File folder	
E Pictures	TestApp_Memory_microblaze_0	8/4/2011 5:26 PM	File folder	
Judeos	ml506_bsb_system.bsb	5/3/2010 10:59 AM	BSB File	
	ml506_bsb_system.mhs	5/12/2010 12:35 PM	MHS File	
🖏 Homegroup	ml506_bsb_system.mss	5/12/2010 12:35 PM	MSS File	
	ml506_bsb_system	5/12/2010 12:43 PM	Xilinx Platform Stu	
🖳 Computer				
🏭 Local Disk (C:)				
👝 Storage (D:)				
📭 Network				
	•	111		

Then copy these files and put them in your xps folder created earlier:

Open up Xilinx XPS and cancel the first prompt:

BSB	Base System Builder wizard (recommended)
ř.	Blank XPS project
P	Open a <u>r</u> ecent project
Browse	e for More Projects

Click File \rightarrow Open Project.. and browse to the ml506_bsb_system.xmp file in the xps folder and click Open:

			00	
computer bootloops data etc implementation microblaze_0 pcores ready_for_download sw TestApp_Memory_microblaze_0 mI506_bsb_system.xmp				
	data etc implementation microblaze_0 pcores ready_for_download sw TestApp_Memory_microblaze_0	data etc implementation microblaze_0 pcores ready_for_download sw TestApp_Memory_microblaze_0	data etc implementation microblaze_0 pcores ready_for_download sw TestApp_Memory_microblaze_0	data etc implementation microblaze_0 pcores ready_for_download sw TestApp_Memory_microblaze_0

Since this Xilinx project was create with EDK version 12.1 and we are using version 13.2, Xilinx will ask us to update the project. Click Yes:

Platfo	orm Studio
?	This project was created with an older version of EDK. XPS will update the project to current release.
	Select 'Yes' to have XPS do the following changes: 1. Back up your project files to the 'revup' directory. 2. Update your project files to the current version.
	Continue?
	Yes <u>N</u> o

Click next on this first screen:

Version Management Wizard	8 X
This Wizard will help you upgrade the con your project.	res in
Some cores were updated in the repository since your project was last processed. outlines the modifications, offers to automatically upgrade to the current backward revision, or provides more information about upgrading to the current revision of the	-compatible
ATTENTION Changes will be made to the MHS file. You can cancel this wizard at any time with modifying the file, but you might not be able to process your project with the cur of the XPS. The following pages show the cores and their upgraded versions. Click the links for view its data sheet and change log.	rent version
To continue, click Next.	
More Info	Cancel

Click Next on this window and the rest that follow:

ne	eds to make essentia		ese cores. Some of t	d in the repository. XPS he cores might be substitu
Uy	Core Name	Current Version	New Version	Reason for Update
1	microblaze	7.30.a	<u>7.30.b</u>	REMOVED
2	mpmc	6.00.a	<u>6.04.a</u>	REMOVED

Finally, click Finish:

Version Management Wizard	2	×
Apply Changes to your Project IMPORTANT: When you click Finish, XPS modifies the MHS file in your project. This w create a backup copy of the MHS file before the project is modified. No changes will be made to other files in the project.	izard wil	II
The following files will be modified: system.mhs		
The following changes will be made: Core microblaze 7.30.a will be replaced by 7.30.b Core mpmc 6.00.a will be replaced by 6.04.a Core plb_v46 1.04.a will be replaced by 1.05.a Core clock_generator 4.00.a will be replaced by 4.02.a The following changes: Core microblaze 7.30.a needs to be replaced by 8.20.a Core lmb_v10 1.00.a needs to be replaced by 2.00.b Core lmb_bram_if_cnthr 2.10.b needs to be replaced by 3.00.b Core mdm 1.00.g needs to be replaced by 2.00.b Core proc_sys_reset 2.00.a needs to be replaced by 3.00.a		
More Info	Can	cel

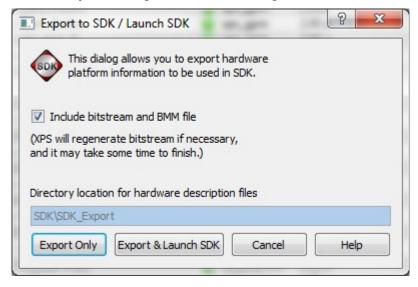
After it finishes loading the files take a look at the peripherals. Notice that it has an ac97 controller:

Name	Bus Name	IP Туре	IP Version	
dlmb		🛕 lmb_v10	1.00.a	
ilmb		🛕 lmb_v10	1.00.a	
opb_v20_0		🛕 opb_v20	1.10.d	
mb_plb		📩 plb_v46	1.05.a	
🖌 🗄 microblaze	0	🛕 microblaze	7.30.b	
🕀 lmb_bram		📩 🙀 bram_block	1.00.a	
💷 🖶 dlmb_cntlr		🛕 lmb_bram_i.	. 2.10.b	
🕀 ilmb_cntlr		🛕 lmb_bram_i.	. 2.10.b	
		📩 📩 xps_mch_e	3.01.a	
DDR2_SDR	AM	📩 mpmc	6.04.a	
🖶 plbv46_opt)	plbv46_opb.	. 1.01.a	
← mdm_0		🛕 mdm	1.00.g	
xps_intc_0		📩 xps_intc	2.01.a	
😐 opb_ac97_o		🕘 opb_ac97_c.	. 1.00.a	
xps_epc_0		🙀 xps_epc	1.02.a	
DIP_Switch	e	📩 xps_gpio	2.00.a	
💷 😥 🕀 🛄 💷		🚖 xps_gpio	2.00.a	
EDs Posit	i	🕁 xps apio	2.00.a	

Click on Device Configuration \rightarrow Update Bitstream (this can take up to 30min):

😌 Xilinx P	latform St	tudio (EDI	(_0.61xd)	- C:\A	C97sine\xps\ml506_	_bsb_syste	m.xmp - [System As:
🔶 File 🛛	Edit Vie	w Proje	ct Hard	ware	Device Configurat	tion Deb	oug Simulation \
5	7	🛛 🕹	88 🔡	BRA	聞部 Update Bitstre	am	
IP Catalog					🖀 Download Bits	stream	↔◻♂×
1: 🕀							M
B 1.11			10	-			B

After this finishes click Project \rightarrow Export Hardware Design to SDK and click Export Only:



Xilinx SDK

Close Xilinx XPS and open Xilinx SDK.

Choose the workspace to be the sdk folder we created earlier and click OK

Workspace	e Launcher			×
	orkspace cores your projects in a folder called a workspace. orkspace folder to use for this session.			
<u>W</u> orkspace:	C:\AC97sine\sdk		-	Browse
🔲 <u>U</u> se this a	s the default and do not ask again	ОК		Cancel

Click File \rightarrow New \rightarrow Xilinx C Project.

Xilinx will ask you to Specify a hardware platform, click Specify:

👀 No H	lardware Platforms in the Workspace
?	SDK requires a hardware platform specification to support application development. The hardware platform specification is exported by either Project Navigator or Xilinx Platform Studio. Currently, no hardware platforms have been brought into the workspace. Select 'Specify' to specify one now, or 'Cancel' to cancel the wizard.
	Specify Cancel

Name the project ac97sine and browse to the exported files we created from Xilinx XPS and click Finish:

🔞 New Hardwa	re Project	
New Hardwa Create a new H	are Project Hardware Project.	
Project name:	ac97sine	
✓ Use <u>d</u> efau	It location	
Location: C:	AC97sine\sdk\ac97sine	Browse
Cho	ose file system: default 💌	
This file usua The specifica	bath to the hardware specification file exported from Project Navigato ally resides in SDK/SDK_Export/hw folder relative to the XPS project loc ation file and associated bitstream content will be copied into the wor e\xps\SDK\SDK_Export\hw\ml506_bsb_system.xml	ation.
➡ Bitstream	and BMM Files	
	you'd like to associate a bitstream different than the one that was exp ify that below.	orted from XPS,
Bitstream:	C:\AC97sine\xps\SDK\SDK_Export\hw\ml506_bsb_system.bit	Browse
BMM File:	C:\AC97sine\xps\SDK\SDK_Export\hw\ml506_bsb_system_bd.bmm	Browse
?	<u> </u>	Cancel

Name the C project 'play_sine' and choose the Empty Application Template, then click Next:

🐼 New Project			
New Xilinx C Proje Create a managed ma		Choose from one of the sample applications.	G
Project name: play_s	ine		
✓ Use <u>d</u> efault location	on		
Location: C:\AC97sir	ne\sdk\play_sine		Browse
Choose file	system: default 💌		
Hardware Platform:	ac97sine		-
Processor:	microblaze_0		_
Dhrystone Empty Application Hello World IwIP Echo Server Memory Tests Peripheral Tests SREC Bootloader Xilkernel POSIX Thre	eads Demo	Description A blank C project.	
?		< <u>B</u> ack Next > Einish	Cancel

Name the board support package 'play_sine_bsp' and click Finish:

🛞 New Project		
New Xilinx C Create a mana	Project aged make application project. Choose from one of the sample applications.	G
	w Board Support Package project late provided by application 'Empty Application' will be used to configure the pr	roject.
Project na	me: play_sine_bsp	
🔽 Use de	efault location	
Location:	C:\AC97sine\sdk\play_sine_bsp	Browse
	Choose file system: default 💌	
	Support Packages found	
?	< <u>B</u> ack <u>N</u> ext > Finish	Cancel

The files I will be using can be found in:

ml506_std_ip_pcores\sw\standalone

Right-click the 'src' folder under the play_sine project and click New \rightarrow Source File:

쀁 Project Explo	orer 🛛 🗖 🗖	system.xml	system.mss 🛛
a 🗇 ac97sine		play_sine_b	osp Board Support Package
syste syste syste	em_bd.bmm em.bit em.xml	Modify this BSF	
⊿ 😂 play_sin ⊳ 👘 Inclu			ort Package is compiled to run on the fo
4 🔁 SI	New	· · · ·	Project
	Go Into		File
⊳ 🏙 play_	Open in New Window		File from Template
	Сору	Ctrl+C	😂 Folder
ti i	Paste	Ctrl+V	Class
3	C Delete	Delete	h Header File
	Move		C Source File
	Rename	F2	Source Folder
2	Import		C Project
2	Export		C++ Project
Į.] Refresh	F5	Ctrl+N
	Index	+	_SDRAM mpmc <u>Documentation</u>
	Make Targets	+	:hes_8Bit gpio <u>Documentation</u>
	Resource Configurations	+	net_MAC litemac <u>Documentation</u>
			EEPROM iic Documentation

Name the file 'play_sine.c' and click Finish:

New Source	File		
Source File Create a new s	source file.		
Source fol <u>d</u> er:	play_sine/src		<u>B</u> rowse
Source fil <u>e</u> :	play_sine.c]	
<u>T</u> emplate:	Default C source template	•	Configure
?		Finish	Cancel

Delete all the auto-generated comments and put this in (copy and paste this code). The original version can be found in ml506_sdt_ip_pcores\sw\standalone\test_ac97. I cleaned it up and added the ability to choose different frequencies and use the LCD screen (its about 4 ¹/₂ pages long):

```
#include <stdio.h>
#include <stdio.h>
#include <math.h>

#include "xio.h"
#include "xuartns550_1.h"
#include "sleep.h"
#include "sleep.h"
#include "lcd.h"
#include "lcd.h"
#include "memory_map.h"
#define UART_CLOCK XPAR_XUARTNS550_CLOCK_HZ
#if !SIM
#define UART_BAUDRATE 9600 /* real hardware */
#else
#define UART_BAUDRATE (UART_CLOCK / 16 / 3) /* simulation */
#endif
```

/* local prototypes */ volatile int data in buf len; int data in buf[19]; volatile int dummy; volatile int cur sound len; volatile unsigned char *cur sound ptr; unsigned int sound ptr; int c; int freq = 22; #defineMY_AC97_BASEADDRXPAR_OPB_AC97_CONTROLLER_REF_0_BASEADDR#defineAC97_InFIFOMY_AC97_BASEADDR#defineAC97_OutFIFOMY_AC97_BASEADDR + 0x4#defineAC97_FIFO_StatusMY_AC97_BASEADDR + 0x8#defineAC97_ControlMY_AC97_BASEADDR + 0xC#defineAC97_RegAddrMY_AC97_BASEADDR + 0x10#defineAC97_RegReadMY_AC97_BASEADDR + 0x14#defineAC97_RegWriteMY_AC97_BASEADDR + 0x14#defineAC97_InFIFO_Full0x01#defineAC97_InFIFO_Full0x02#defineAC97_OutFIFO_Full0x04 #define AC97_OutFIFO_Full 0x04
#define AC97_OutFIFO_Empty 0x08 #define AC97_Reg_Access Finished 0x10 #defineAC97_CODEC_RDY0x20#defineAC97_REG_ACCESS0x40 #define AC97_Enable_In_Intr 0x01 // AC97 CODEC Registers 0x00 #define AC97 Reset **#define** AC97 MasterVol 0x02 #define AC97_Mastervor #define AC97_HeadphoneVol 0x04 0x06 0x08 **#define** AC97 MasterVolMono **#define** AC97_Reserved0x08 0x0A #define AC97_PCBeepVol **#define** AC97 PhoneInVol 0x0C #define AC97_MicVol 0x0E #define AC97 LineInVol 0x10 #define AC97 CDVol 0x12 #define AC97_VideoVol 0x14 #define AC97_AuxVol 0x16 #define AC97 PCMOutVol 0x18 #define AC97_RecordSelect 0x1A **#define** AC97 RecordGain 0x1C #defineAC97_Reserved0x1E0x1E#defineAC97_GeneralPurpose0x20 #define AC97_3DControl #define AC97_PowerDown 0x22 0x26 #define AC97 ExtendedAudioID 0x28 #define AC97 ExtendedAudioStat 0x2A #define AC97 PCM DAC Rate0 0x78 **#define** AC97 PCM DAC Rate1 0x7A **#define** AC97_Reserved0x34 0x34 **#define** AC97_JackSense 0x72 **#define** AC97 SerialConfig 0x74

```
#define AC97 MiscControlBits
                              0x76
#define AC97 VendorID1
                               0x7C
#define AC97 VendorID2
                                0x7E
// Volume Constants
#define AC97 VolMute 0x8000
#define AC97 VolMax 0x0000
#define AC97_VolMin 0x3F3F
#define AC97_VolMid 0x1010
void WriteAC97Reg( int reg addr, int value) {
 XIo Out32 (AC97 RegWrite, value);
 XIo Out32 (AC97 RegAddr, reg_addr);
  usleep (10000);
}
int ReadAC97Reg( int reg addr) {
 XIo Out32 (AC97 RegAddr, reg addr | 0x80);
 usleep (10000);
 return XIo In32(AC97 RegRead);
}
void init sound() {
 printf("Initializing AC97 CODEC...\r\n");
 // reset all reg's to known states (cause MUTE to all)
 WriteAC97Reg(AC97 Reset,0);
 usleep (1000);
 while (!(XIO In32(AC97 FIFO Status) & AC97 CODEC RDY)) {};
  xil printf ("PowerDown Reg State (Should be 0x000F) = x \ln^{r}, ReadAC97Reg
(AC97 PowerDown));
  // turn on external amp
  xil printf ("Turning on External Power Amp \n\r");
 WriteAC97Reg(AC97 JackSense, 0x3F00); // set Jack Sense Pins
  // powerdown DAC and ADC temporarily
 WriteAC97Reg(AC97 PowerDown, 0x0300);
 usleep (1000000);
  // initialize LCD and write to it
 LCDOn();
  LCDInit();
 LCDPrintString ("1. 400Hz 2. 1kHz", "3. 5kHz");
 xil printf ("\n\r\n\rPick the frequency:\n\r");
  xil printf ("1. 400Hz\n\r2. 1kHz\n\r3. 5kHz\n\r\n\r\n\r");
  while (!(XUartNs550 GetLineStatusReg(UART BASEADDR) & 0x1));
  do {freq = XUartNs550 RecvByte(UART BASEADDR); }
  // check for invalid choices
  while (((char) freq != '1') && ((char) freq != '2') && ((char) freq != '3'));
  if ((char) freq == '1') freq = 110;
  if ((char) freq == '2') freq = 44;
  if ((char) freq == '3') freq = 9;
```

WriteAC97Reg(AC97 ExtendedAudioStat,1); // Enabling VRA mode WriteAC97Reg(AC97 PCM DAC Rate1, 48000); // sampling rate PLAY WriteAC97Reg(AC97 PCM DAC Rate0, 48000); // sampling rate rec xil printf ("DAC sample rate= %d <u>hz</u>\n\r", ReadAC97Reg (AC97 PCM DAC Rate1)); xil printf ("ADC sample rate= %d <u>hz</u>\n\r", ReadAC97Reg (AC97 PCM DAC Rate0)); xil_printf ("vendor id 1= %x\n\r", ReadAC97Reg (AC97_VendorID1)); xil printf ("vendor id 2= %x\n\r", ReadAC97Reg (AC97 VendorID2)); // Turn back on power to ADC and DAC WriteAC97Reg(AC97 PowerDown,0x0000); usleep (100000); xil printf ("PowerDown Reg State (Should be 0x000F) = $x \ln^r$, ReadAC97Reg (AC97 PowerDown)); xil printf ("Misc Control Bits = %x\n\r", ReadAC97Reg (AC97 MiscControlBits)); XIo Out32 (AC97 Control, 0x0000003); // clear FIFOs XIo_Out32(AC97_InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIo_Out32(AC97_InFIFO, 0); XIo_Out32(AC97_InFIFO, 0); XIo_Out32(AC97_InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32 (AC97 InFIFO, 0); XIo_Out32(AC97_InFIFO, 0); XIo_Out32(AC97_InFIFO, 0); XIO Out32(AC97 InFIFO, 0); XIO Out32(AC97 InFIFO, 0); // turn off digital loopback WriteAC97Reg(AC97 GeneralPurpose,0x0000); xil printf ("General Purpose reg state = %x\n\r", ReadAC97Reg (AC97 GeneralPurpose)); WriteAC97Reg(AC97 SerialConfig,0x7000); xil printf ("config reg state = %x\n\r", ReadAC97Reg (AC97 SerialConfig)); WriteAC97Reg(AC97 MasterVol, AC97 VolMid); WriteAC97Reg(AC97_HeadphoneVol, AC97_VolMid); WriteAC97Reg(AC97 MasterVolMono, AC97 VolMid); WriteAC97Reg(AC97_PCBeepVol,AC97_VolMute);WriteAC97Reg(AC97_PCBeepVol,AC97_VolMute);WriteAC97Reg(AC97_PhoneInVol,AC97_VolMute);WriteAC97Reg(AC97_CDVol,AC97_VolMute);WriteAC97Reg(AC97_VideoVol,AC97_VolMute);WriteAC97Reg(AC97_AuxVol,AC97_VolMute);WriteAC97Reg(AC97_PCMOutVol,AC97_VolMute);WriteAC97Reg(AC97_RecordSelect,0x0000);

```
WriteAC97Reg(AC97 MicVol,
                                   0x0040);
 WriteAC97Reg(AC97 LineInVol, AC97 VolMid);
}
void play sound() {
  int i;
  int j;
  xil printf ("Play Start\n\r");
  sound ptr = DDR BASEADDR;
  WriteAC97Reg(AC97 RecordGain, AC97 VolMute);
 WriteAC97Reg(AC97 PowerDown, 0x0100);
 WriteAC97Reg(AC97 LineInVol,
                                AC97 VolMute);
  i = 0;
  do {
    XIo Out32 (AC97 Control, 0x0000003); // clear FIFOs
    XIo_Out32(AC97_InFIFO, 0);
    XIO Out32(AC97 InFIFO, 0);
   XIO Out32(AC97 InFIFO, 0);
    XIO Out32(AC97 InFIFO, 0);
   XIO Out32(AC97 InFIFO, 0);
   XIO_Out32(AC97_InFIFO, 0);
XIO_Out32(AC97_InFIFO, 0);
    XIo_Out32(AC97_InFIFO, 0);
  } while (XIo_In32(AC97 FIFO Status) & 0x0040);
  while (1) {
      if (XIo In32(AC97 FIFO Status) & 0x0040) xil printf ("play underrun\n\r");
      while (XIo_In32(AC97_FIF0_Status) & AC97_InFIF0_Half_Full) {};
    // The frequency is 44.1kHz so if we divide 44.1kHz by 22 we
    // get 2004.54 ~ 2kHz sine wave
    switch ((i) % freq) {
       case 0: j = 0; break;
       case 1: j = 9232; break;
       case 2: j = 17715; break;
       case 3: j = 24764; break;
       case 4: j = 29806; break;
       case 5: j = 32434; break;
       case 6: j = 32435; break;
       case 7: j = 29808; break;
       case 8: j = 24766; break;
       case 9: j = 17718; break;
       case 10: j = 9234; break;
       case 11: j = 3; break;
       case 12: j = -9229; break;
       case 13: j = -17713; break;
       case 14: j = -24762; break;
       case 15: j = -29805; break;
       case 16: j = -32434; break;
       case 17: j = -32435; break;
       case 18: j = -29809; break;
       case 19: j = -24768; break;
       case 20: j = -17720; break;
       case 21: j = -9237; break;
     ļ
   i = i+1;
```

```
XIo_Out32(AC97_InFIFO, j);

int main ()
{
  XUartNs550_SetBaud(UART_BASEADDR, UART_CLOCK, UART_BAUDRATE);
  XUartNs550_SetLineControlReg(UART_BASEADDR, XUN_LCR_8_DATA_BITS);
  init_sound();
  while (1) {
    play_sound();
    }
    return 0;
}
```

After pasting in the above code, your window should look something like this:

```
🖻 play_sine.c 🛛
                                                                                          *
   #include <stdio.h>
   #include <math.h>
                                                                                         Ξ
   #include "xio.h"
   #include "xuartns550 1.h"
   #include "sleep.h"
   #include "xparameters.h"
   #include "lcd.h"
   #include "memory_map.h"
   #define UART_CLOCK XPAR_XUARTNS550_CLOCK_HZ
   #if !SIM
   #define UART BAUDRATE 9600
                                                                          /* real h
   #else
   #define UART BAUDRATE (UART CLOCK / 16 / 3) /* simula
   #endif
   /* local prototypes */
   volatile int data in buf len;
   int data in buf[19];
   volatile int dummy;
   volatile int cur sound len;
   volatile unsigned char *cur sound ptr;
   unsigned int sound ptr;
   int c;
   int freq = 22;
   #defineMY_AC97_BASEADDRXPAR_OPB_AC97_CONTROLLEF#defineAC97_InFIFOMY_AC97_BASEADDR#defineAC97_OutFIFOMY_AC97_BASEADDR + 0x4#defineAC97_FIFO_StatusMY_AC97_BASEADDR + 0x8#defineAC97_ControlMY_AC97_BASEADDR + 0x8#defineAC97_RegAddrMY_AC97_BASEADDR + 0x10#defineAC97_RegReadMY_AC97_BASEADDR + 0x10#defineAC97_RegReadMY_AC97_BASEADDR + 0x14#defineAC97_RegWriteMY_AC97_BASEADDR + 0x18#defineAC97_InFIFO_Full0x01
   #define AC97_InFIF0_Full
   #define AC97 InFIFO Full 0x01
```

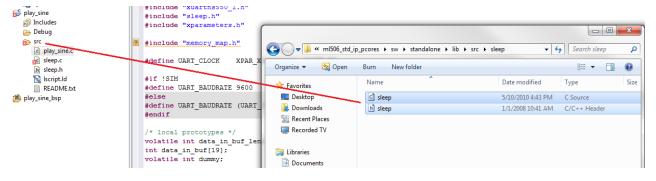
Next we need to add some header files.

Go to the standalone folder:

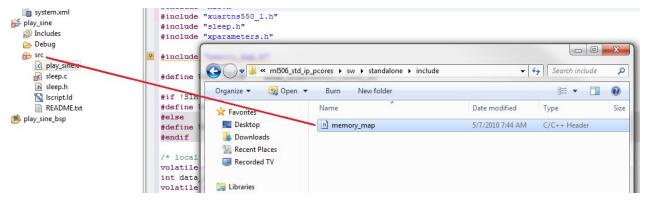
ml506_std_ip_pcores\sw\standalone

and open the ml506_std_ip_pcores\sw\standalone\lib\src\sleep file.

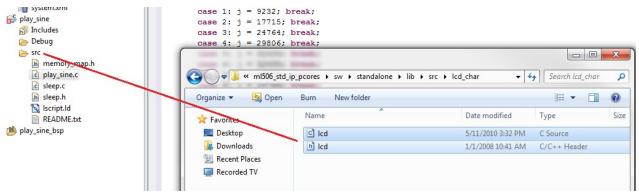
Highlight sleep.c and sleep.h and drag it into the src folder:



Next, go to the $m1506_std_ip_pcores\sw\standalone\include folder and drag the memory_map.h file into the src folder:$



Then, go to the ml506_std_ip_pcores\sw\standalone $lib\src\lcd_char$ folder and drage lcd.h and lcd.c into the src folder:



Click Save and your project should start building and compiling.

Right-click on the play_sine project and click Generate Linker Script, accept the default settings and click Generate overwriting existing files if asked:

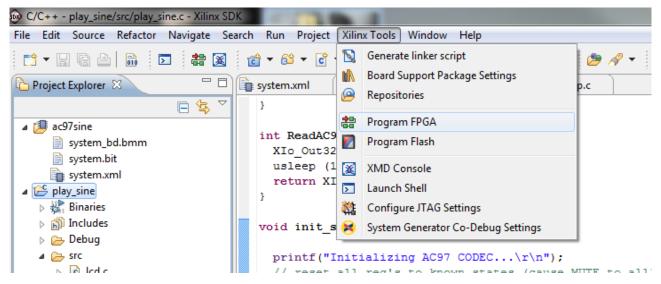
Generate a linker script							-2
Generate linker script							
Control your application's memory m	ap.						
				Basic Advanced			
Project: play_sine					Think with allock and	4-	
Output Script:				Place Code Sections in:	ilmb_cntlr_dlmb_cnt	ır	-
C:\AC97sine\sdk\play_sine\src\lscrip	ot.ld		Browse	Place Data Sections in:	ilmb_cntlr_dlmb_cnt	lr	-
Modify project build settings as follow	ws:			Place Heap and Stack in:	ilmb_cntlr_dlmb_cnt	lr	•
Set generated script on all project bu	ild configurations		•	Heap Size:	1 KB		
				Stack Size:	1 KB		
Memory	Base Address	Size					
ilmb_cntlr_dlmb_cntlr	0x00000000	64 KB					
SRAM_MEM0_BASEADDR	0x8D000000	1 MB					
SRAM_MEM1_BASEADDR	0x8E000000	32 MB					
DDR2_SDRAM_MPMC_BASEADDR	0x90000000	256					
Fixed Section Assignments							
?						Generate	Cancel
•						Generate	Cancel

Program the FPGA

Turn on the ML506 and open up a HyperTerminal or TeraTerm etc... I use putty here:

😵 PuTTY Configuration		×
Category:		
	Basic options for your PuTT	(session
Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation	Specify the destination you want to con- Serial line COM1 Connection type: Raw O Telnet Rlogin S Load, save or delete a stored session Saved Sessions	S <u>p</u> eed 9600
Selection Colours Data Proxy Telnet Rlogin	Default Settings	Load Sa <u>v</u> e Delete
⊞·· SSH Serial	Close window on e <u>x</u> it: ⊚ Always ⊚ Never	n clean exit
About	Open	<u>C</u> ancel

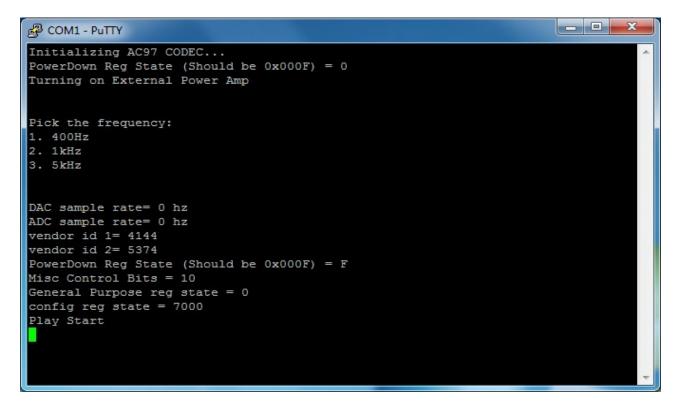
Then in SDK click Xilinx Tools \rightarrow Program FPGA.



Make sure the bistream and BMM files point to the files you exported from XPS, then set your play sine.elf file to be initialized onto the block RAM then click Program:

	🐵 Program FPGA						
	Program FPGA Specify the bitstream and the ELF files that reside in BRAM memory						
	Hardware S	pecifica	tion: C:\AC97sine\sdk\ac97sine\system.xml				
1	Bitstream:	C:\AC	97sine\xps\SDK\SDK_Export\hw\ml506_bsb_system.bit	Browse			
	BMM File:	C:\AC	97sine\xps\SDK\SDK_Export\hw\ml506_bsb_system_bd.bmm	Browse			
	Processor		ELF File to Initialize in Block RAM				
	microblaze	e_0	C:\AC97sine\sdk\play_sine\Debug\play_sine.elf 🗾 🔫				
	?		Program	Cancel			

The FPGA is now programmed. Look to the terminal window for the directions and choose your sine wave by typing on your PC keyboard:



Some instructions are also displayed on the LCD:



SDK will create a download.bit file located in : C:\AC97sine\sdk\ac97sine. I suggest you rename this to something like playsine.bit

Programming your FPGA with the Compact Flash

Summary

You can save your programs on the compact flash so that when you turn on the ML506 it will automatically run your program without having to open up iMPACT or Xilinx SDK to program it. To do this we need the bitstream for your project and then we convert it to a .ace file. Then we save this file into one of the configuration folders located in the file system of the compact flash. We then set the configuration mode on SW3 to correspond to the folder number we saved our .ace file into and turn on the board.

We will use the bitsream created from the play sine wave tutorial. It is located in(on my computer):

SDK names the file download.bit but I renamed it to playsine.bit

Creating an .ace file

This tutorial comes directly from this website: <u>http://www.fpgadeveloper.com/2009/10/convert-bit-files-to-system-ace-files.html</u> I will summarize it here for convenience:

First add the Xilinx command: xmd

to your system path (google 'how to add to path windows 7' if you don't know how):

To get to this window right-click Computer, click Advanced system settings Environment Variables.., Edit the variable called 'Path' under System variables:

Edit System Variab	le
Variable <u>n</u> ame:	Path
Variable <u>v</u> alue:	<pre>/stem\;C:\Xilinx\13.2\ISE_DS\EDK\bin\nt64</pre>
	OK Cancel

nclude in library 🔻 🛛 Share with ୟ	- Burn	New folder	
Name		Date modified	Туре
dev dev		7/4/2011 9:21 AM	File folder
eclipse		7/30/2011 9:53 AM	File folder
Fraps		8/4/2011 1:30 PM	File folder
MATLAB		7/14/2011 8:53 AM	File folder
b ml506-edk-13-2		8/3/2011 6:36 PM	File folder
ModelSimTestbenches		6/30/2011 7:10 PM	File folder
🎍 PerfLogs 鷆 Photoshop 鷆 Program Files	Folders: Files: eq	work Ltestbench.cr.mti, eq_t 8/4/2011 8:22 AM	estbench.mpf, gre File folder
			File folder File folder
Program Files (x86) Simulink_Programs		7/30/2011 1:05 PM 5/21/2011 10:44 PM	File folder
starcraftInstall		5/29/2011 10:44 PM	File folder
testbench		7/13/2011 1:54 PM	File folder
Users		7/12/2011 12:50 PM	File folder
Windows		8/1/2011 7:31 PM	File folder
WTablet		8/2/2011 3:13 PM	File folder
Xilinx		8/1/2011 4:42 PM	File folder
			F11 6 1 1
JiinxML506		8/4/2011 5:23 PM	File folder

Create a folder on your local drive called: SysACE

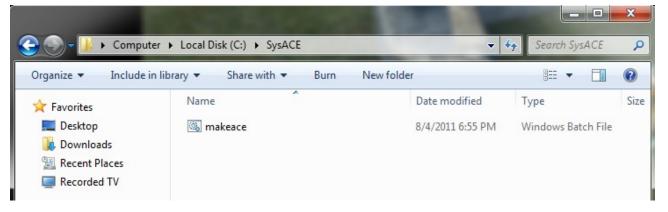
Open up a text editor and copy and paste this code:

```
@echo off
if "%1" == "" goto error
xmd -tcl ./genace.tcl -jprog -hw %1.bit -board ml505 -ace my_%1.ace
goto end
:error
echo Makeace - by FPGA Developer http://www.fpgadeveloper.com
echo.
echo Usage: makeace bitfile (without .bit extension)
echo Example: makeace project
:end
echo.
```

						~~
Save As						X
	omputer 🕨 Local Disk (C:) ► SysACE	- ↓	Search SysACE		٩
Organize 🔻 Ne	w folder					?
Documents	 Name 	^		Date modified	Туре	
J Music						
Pictures		No item:	s match you	ir search.		
😽 Videos						
 Homegroup Computer Local Disk (C:) Storage (D:) Network 						4
File <u>n</u> ame:	makeace.bat					-
Save as <u>t</u> ype:	All Files					_
Aide Folders	<u>E</u> ncoding:	ANSI	-	Save	Cancel	

Then save the file as makeace.bat into the SysACE folder:

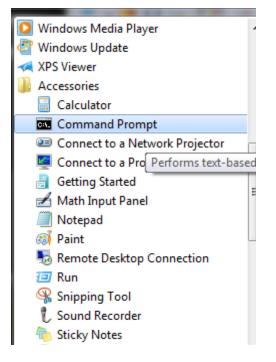
Now you should have this file in you SysACE folder:



Now we need the bitstream of our project we want to convert to an ACE file. Lets use the playsine.bit from the previous tutorial (if you didn't rename it, it would be called download.bit). Copy and paste this file into the SysACE folder:

😪 🌍 🗕 📔 🕨 Compute	r ▶ Local Disk (C:) ▶ SysACE	▼ 4 ₇	Search SysACE 👂
Organize 👻 📄 Open	Burn New folder		III 🔹 🚺 🔞
☆ Favorites ■ Desktop ↓ Downloads	Name Mame Makeace Name		Type Size Windows Batch File
Recent Places Recorded TV	E PARTIE		uter 🕨 Local Disk (C:) 🕨 Ad
 Libraries Documents Music Pictures 		Organize Op	Name
🛃 Videos 🤣 Homegroup		Downloads Recent Places Recorded TV	 playsine.bit system.bit system system system_bd.bmr

Open up a windows command prompt:

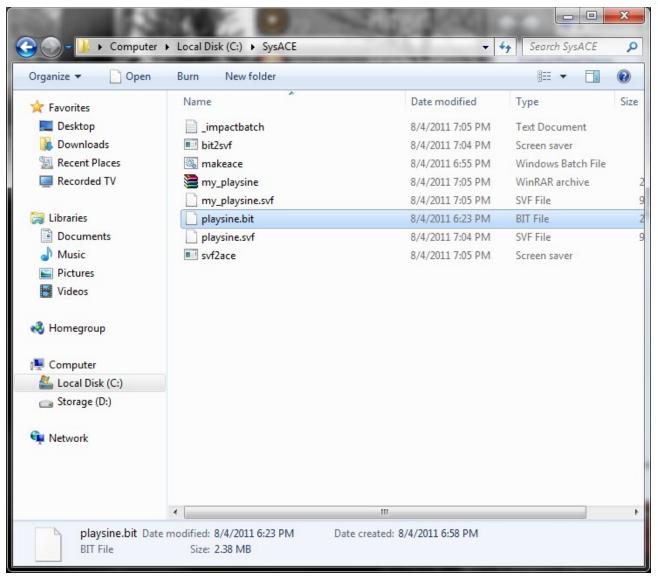


Type these commands to change directory to the SysACE folder:

cd \SysACE

Then to convert the file type: makeace playsine

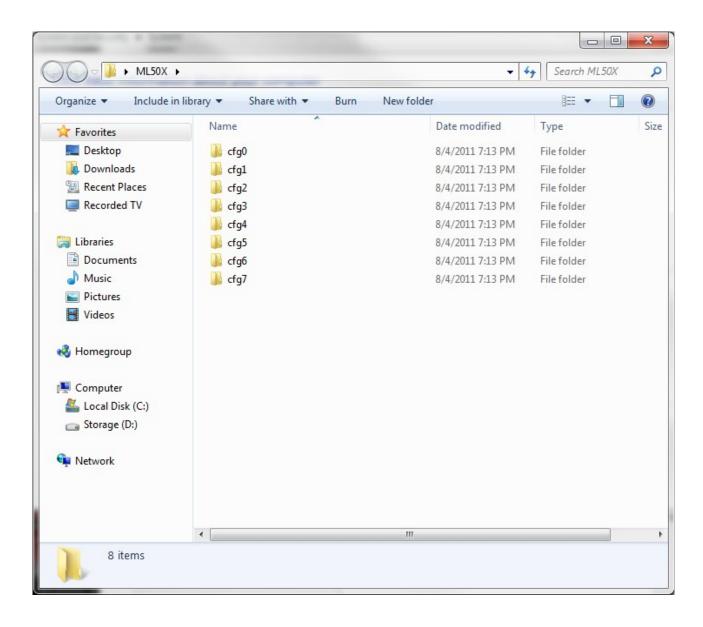
Once this finishes, several files will be created in the SysACE folder, we are interested in the my_playsine.ace file:

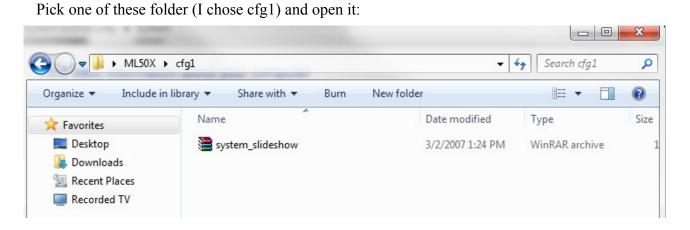


Setting up the Compact Flash

Make sure that your compact flash is properly formatted, see the beginning of this entire tutorial.

Turn of the ML506 and gently remove the compact flash. Insert the compact flash into your compact flash reader and open up the ML50X folder:





Inside you will notice some factory files, delete this file and replace it with the my_playsine.ace file we created (note: deleting this file will render the Xilinx Demo program inoperable):

Organize 👻 🚺 Open 👻	Share with 🔻 🛛 Burn 🛛 New folder		≣ • 🔳	?
🔆 Favorites	Name	Date modified	Туре	Siz
 Desktop Downloads Recent Places Recorded TV 	my_playsine	8/4/2011 7:05 PM	WinRAR archive	

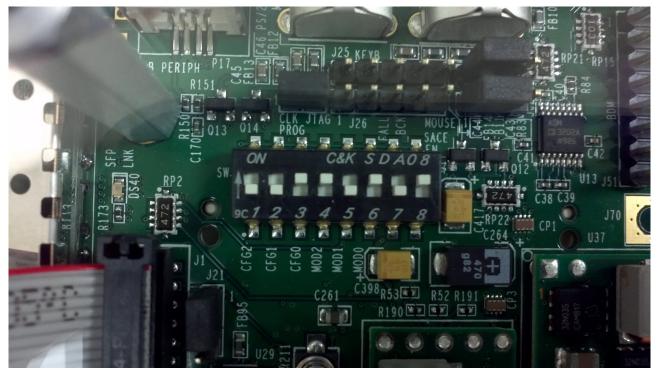
Safely eject, the compact flash:

	:	
Safely R	emov	e Hardware and Eject Media
to 🕨		

Then insert the compact flash back into the ML506.

Since our file is in cfg1, we need to change the first three bits of SW3 to : 001 which is 1 in binary and leave the rest of the bits as: 10101.

Here is a picture:



Now turn on the board and your program should start automatically.

You can put more designs into the other folders, just change the first three bits of SW3 to correspond to the folder number, then turn on the board.

Conclusion

More Designs

By finishing this tutorial, you should be familiar with the many capabilities of the ML506 and the various Xilinx softwares such as ISE and EDK. There are plenty of tutorials online and from Xilinx. Here are some links if you want to learn more:

System Generator: <u>http://www.xilinx.com/support/sw_manuals/sysgen_user.pdf</u> Xilinx EDK: <u>http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_1/edk_ctt.pdf</u> FPGA Tutorials: <u>http://www.fpgadeveloper.com/</u>