Dünner Kirchweg 77
32257 Bünde
Germany
www.trenz-electronic.de

Spartan-II Development System

2001-October-24

Application Note: SoC with XR16 RISC CPU

Introduction

The Spartan-II Development System is designed to provide a simple yet powerful platform for FPGA development, which can be easily expanded to reflect your application's requirements. The following application note was developed to give the engineer a quick hands-on experience on System-on-Chip design with FPGAs.

CPUs are the central component of all System-on-Chip designs. While most designs are baed upon FPGA versions of existing 8bit architectures, this application note utilizes Jan Gray's *XR16* CPU, which implements a streamlined RISC concept. Unlike other Open Source designs, the amount of documentation and support material available for the *XR16* CPU is unparalleled, which makes the CPU especially useful in education scenarios. The *XR16*'s core features:

- pipelined RISC with sixteen 16bit registers and 16bit instructions
- 3 stage pipeline (fetch, decode, execute)
- approximately 1.4 cycles per instruction
- 64kB address range
- integrated DMA engine
- interrupt handling
- very efficient FPGA implementation
- C-compiler available

Based on the *XR16* CPU, this application note implements a System-on-Chip with the following features:

- 16bit RISC CPU
- 4kB of firmware ROM/ general purpose RAM
- VGA text display with 64x32 characters
- ASCII character generator

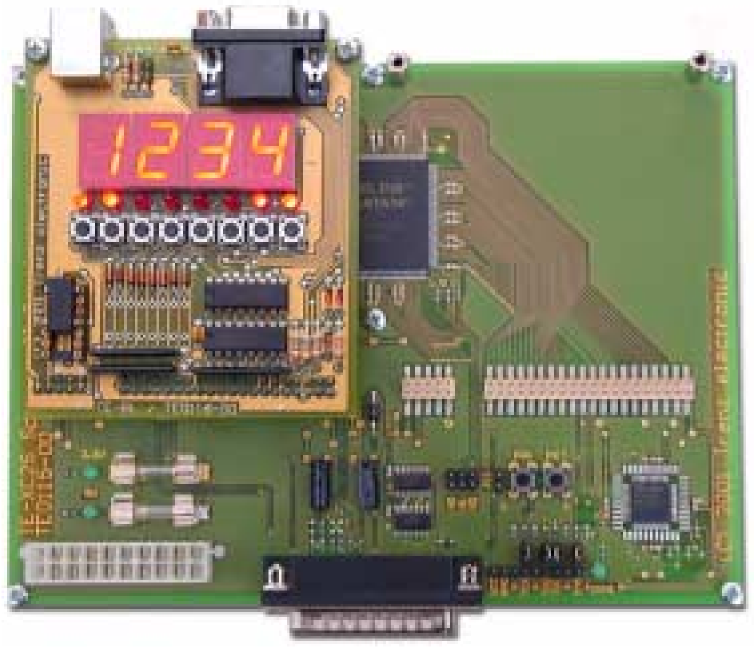

Figure 1: TE-XC2S Base Board with TE-BL Expansion

Design Resources

This application note combines the efforts of several authors. These resources are intellectual property of their authors and copyrighted under the terms of these authors. In case some of these resources are not included in the project archive provided with this application note or to update these resources to their most recent versions, you'll need to download these resources first.

The *XR16* CPU was created by Jan Gray of Gray Research LLC. The XR16 CPU is part of the XSOC project. To review the *XSOC License Agreement* and to download the latest version of the XSOC distribution, go to:

- http://www.fpgacpu.org/xsoc

The *lcc* is a retargetable compiler for ANSI C which is described in the book *A Retargetable C Compiler: Design and Implementation*. This book is a detailed tour of the lcc 3.6 source code. The ftp distribution includes the source code for the complete compiler, the code generators for all its targets, and the code-generator generator that produced them. The compiler may be downloaded from

- http://www.cs.princeton.edu/software/lcc/

Architectural Description

The design is created from several entities, which are described in the following. Refer to Figure 2 for the design hierarchy.

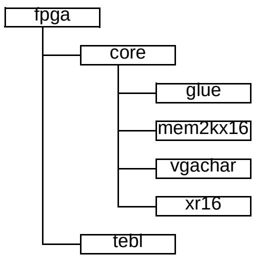

Figure 2: Design hierarchy

Entity core

The entity core implements the actual functionality of the design by combining a set of building blocks. Refer to Figure 3 for a block diagram. The instances serve the following purposes:

- *xr16:* the XR16 CPU
- *mem2Kx16:* the firmware ROM and general purpose RAM
- *vgachar:* the video memory and character generator
- *glue:* the interconnecting glue logic

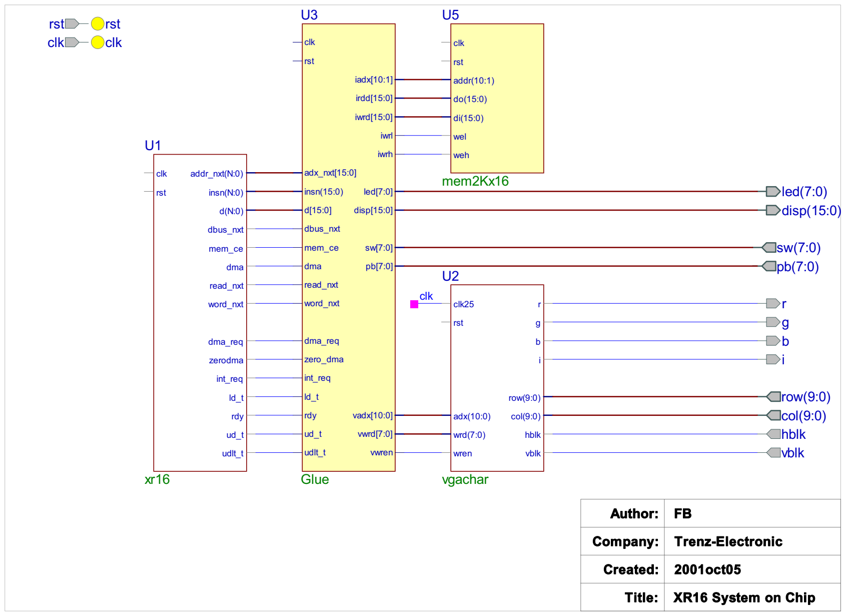

Figure 3: Entity core

Entity XR16

The *XR16* CPU is created in Verilog. As the author of this application note prefers VHDL, this entity is instantiated as a black box.

The black box is resolved during design translation on netlist level. The netlist is created from a distinct Verilog project, implementing the *XR16* CPU from Jan Gray's unaltered sources. See *Figure 4* for the XR16 hierarchy. Synthesis of this project produces a netlist, which is instantiated as a black box in the VHDL project. To do so, the synthesis option *Add I/O Buffers* must be switched off, which prevents the synthesizer from inserting I/O pads into the top level ports. Please note, that this project is just synthesized, but not implemented.

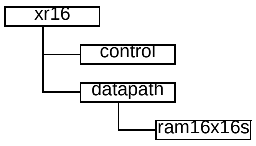

Figure 4: XR16 hierarchy

The CPU, its instruction set and architecture are very well documented, please refer to the various resources provided by Gray Research.

Entity mem2kx16

The entity *mem2Kx16* implements the firmware ROM and general purpose RAM. The memory has a capacity of 4kB, organized as 2k x 16.

The memory uses *BlockRAM* resources, which may be initialized during FPGA configuration to provide ROM functionality. To do so, *INIT_xx* attributes are assigned to the *RAMB4_S8* entities.

As attaching the *INIT_xx* attributes to the RAMs is a cumbersome task, a *perl* script has been created, automating this step. The script takes a *.hex* file from the xr16 assembler and creates a *.vhd* from it, effectively creating a firmware ROM. If you are wondering where to find a perl interpreter: The Xilinx WebPACK tools come with a perl interpreter named *xilperl* located in the *bin\nt* directory.

The firmware writes a text message to the VGA display, reads in the push buttons and switches and displays their values using the LEDs and 7-segment displays. See the following listing:

```
#define TITLE (char*) 0x8000
#define LED   (char*) 0x9000
#define DISP  (short*)0xA000
#define PB    (char*) 0xB000
#define SW    (char*) 0xC000

char msg[] = {"TE-XC2S App Note: SoC
with XR16 RISC CPU"};

void main()
{ char* src;
  char* dst;

  src= msg;
  dst= TITLE;
  while(*src)
  { *dst++ = *src++;
  }

  while(1)
  { *LED=  *SW;
    *DISP= *PB;
  }
}
```

Entity vgachar

The entity *vgachar* implements the video RAM and character generator. The screen is organized in 32 lines with 64 characters each, resulting in 2kB which are again implemented using *BlockRAM* resources.

The character generator creates pixels from 7bit ASCII characters. Each pixel is created from an 8x8 matrix, resulting in 1kB of ROM for the complete character set.

Entity glue

The glue logic interconnects the processor with its peripherals. To do so, it implements the following functionality:

- address decoding
- byte lane switching
- led and 7-segment registers

This is the place where you will most probably want to add and implement your own ideas. To do so, you are highly encouraged to re-engineer these few lines and watch the signals in a behavioral simulator- which is much more enlighting than any description the author could possibly give here.

Entity fpga

The entity *fpga* is the top level of the design. It performs the following functions:

- create a reset signal
- lock all I/Os to their pad locations
- specify timing constraints

Furthermore, this entity instantiates the core logic and the *TE-BL* interface.

This entity uses VHDL attributes to pass implementation constraints to the implementation tools. To learn more about this mechanism, please refer to our *Buttons&Lights* application note.

Entity tebl

The entity tebl encapsulates the specific TE-BL interface functionality. The main tasks of this entity are:

- led multiplexing
- 7-segment decoding
- push button debouncing
- switch emulation
- vga timing generation

For further details on this entity, please refer to our *Buttons&Lights* and our *Game of Life* application notes.

Implementation

The project files for this application note are provided in *WebPACK ISE v3.3* format with all synthesis options set up to achieve a push button flow. The XR16 processor is provided as a precompiled edif netlist, to ease implementation of this application note.

The CPU and VGA logic of this design are clocked with 24MHz. To ensure that the implementation meets these timing requirements, the use of proper timing constraints is absolutely necessary. To answer questions regarding implementation and download of projects, our tutorial *First Steps with WebPACK ISE* is highly recommended.

Table 1 summarizes the resource usage of the complete design implemented in a Xilinx XC2S200 device. Please note, that these figures may vary slightly depending on your implementation tools.

Resource	**Usage**	
Number of Slices	470	19%
Number of Slice Flip Flops	329	6%
Total Number 4 input LUTs	644	13%
Number used as LUTs	593	
Number used as route-thru	3	
Number used as 16x1 RAMs	48	
Number of bonded IOBs	22	15%
Number of Tbufs	136	5%
Number of Block RAMs	14	100%
Number of GCLKs	1	
Number of GCLKIOBs	1	
Total equivalent gate count	243,069	
gate count memory	229,376	
gate count logic	13,693	

Table 1: XC2S200 Resource Usage

References

- *Spartan-II Development System*
  Product Specification
  Trenz Electronic
  September 12, 2001
- *Spartan-II Development System*
  First Steps with WebPACK ISE
  Trenz Electronic
  September 25, 2001
- *Spartan-II Development System*
  Application Note: Buttons & Lights
  Trenz Electronic
  September 27, 2001
- *Spartan-II Development System*
  Application Note: Game of Life
  Trenz Electronic
  September 17, 2001
- *The XSOC Project*
  Gray Research LLC
  www.fpgacpu.org/xsoc
- *FPGA CPU and SoC discussion list*
  groups.yahoo.com/group/fpga-cpu
- *A Retargetable C Compiler:*
  Design and Implementation
  Fraser & Hanson, Addison-Wesley
  1995
- *Usenet newsgroup*
  comp.compilers.lcc

Revisions History

Version	Date	Who	Description
1.0	2001oct23	FB	Created

Table 2: Revisions History