Gisselquist
Technology, LLC

OPENARTY SPECIFICATION

Dan Gisselquist, Ph.D.
dgisselq (at) opencores.org

June 20, 2016

Revision History

Rev.	Date	Author	Description
0.0	6/20/2016	Gisselquist	First Draft

Contents

Figures

Figure Page

Tables

Preface

Dan Gisselquist, Ph.D.

1.

Introduction

The goals of this project include:

1. Use entirely open interfaces

   This means not using the Memory Interface Generator (MIG), the Xilinx CoreGen IP, etc. Further, I wish to use all of Arty's on–board hardware: Flash, DDR3-SDRAM, Ethernet, and everything else at their full and fastest speed(s). For example, the flash will need to be clocked at 100 MHz, not the 50 MHz I've clocked it at before. The memory should also be able to support pipelined 32–bit interactions over the Wishbone bus at a 200 MHz clock. Finally, the Ethernet controller should be supported by a DMA capable interface that can drive the ethernet at its full 100Mbps rate.

2. Run using a 200 MHz clock, if for no other reason than to gain the experience of building logic that can run that fast.

3. Modify the ZipCPU to support an MMU and a data cache, and perhaps even a floating point unit.

4. The default configuration will also include three Pmods: a USBUART, an SDCard, and the GPS Pmod.

I intend to demonstrate this project with a couple programs:

1. A very simple program that runs automatically upon startup that can be used to select from among multiple configurations.

2. NTP Server

3. A ZipOS that can actually load and run programs from the SD Card

2.

Architecture

3.

Software

3.1 Directory Structure

3.2 Zip CPU Tool Chain

3.3 Bench Test Software

3.4 Host Software

- `readflash`: As I am loathe to remove anything from a device that came factory installed, the `readflash` program reads the original installed configuration from the flash and dumps it to a file.
- `wbregs`: This program offers a capability very similar to the PEEK and POKE capability Apple user's may remember from before the days of Macintosh. `wbregs <address>` will read from the Wishbone bus the value at the given address. Likewise `wbregs <address> <value>` will write the given value into the given address. While both address and value have the semantics of numbers acceptable to `strtoul()`, the address can also be a named address. Supported names can be found in `regdefs.cpp`, and their register mapping in `regdefs.h`.
- `ziprun`:
- `zipload`:

3.5 Zip CPU Programs

- `ntpserver`:
- `goldenstart`:

3.6 ZipOS

3.6.1 System Calls

- `int wait(unsigned event_mask, int timeout)`

- `int clear(unsigned event_mask, int timeout)`
- `void post(unsigned event_mask)`
- `void yield(void)`
- `int read(int fid, void *buf, int len)`
- `int write(int fid, void *buf, int len)`
- `unsigned time(void)`
- `void *malloc(void)`
- `void free(void *buf)`

3.6.2 Scheduler

4.

Operation

5.

Registers

There are several address regions on the S6 SoC, as shown in Tbl. 5.1.

Binary Address	Base	Size(W)	Purpose
0000 0000 0000 0000 0001 0000 xxxx	0x00000100	16	Peripheral I/O Control
0000 0000 0000 0000 0001 0001 0yyx	0x00000110	8	Debug scope control
0000 0000 0000 0000 0001 0001 10xx	0x00000118	4	Flash control
0000 0000 0000 0000 0001 0001 11xx	0x00000118	4	RTC control
0000 0000 0000 0000 0001 0010 00xx	0x00000120	4	SDCard controller
0000 0000 0000 0000 0001 0010 01xx	0x00000124	4	Packet Controller
0000 0000 0000 0000 0001 0011 00xx	0x00000130	4	GPS Uart
0000 0000 0000 0000 0001 0011 01xx	0x00000134	4	GPS Clock
0000 0000 0000 0000 0001 0011 1xxx	0x00000138	8	GPS Testbench
0000 0000 0000 0000 0001 010x xxxx	0x00000140	32	Ethernet configuration registers
0000 0000 0000 0000 0001 011x xxxx	0x00000160	32	ICAPE2 Configuration Port
0000 0000 0000 0000 10xx xxxx xxxx	0x00000800	1k	Ethernet TX Buffer
0000 0000 0000 0000 11xx xxxx xxxx	0x00000c00	1k	Ethernet RX Buffer
0000 0000 0000 1xxx xxxx xxxx xxxx	0x00008000	32k	On-chip Block RAM
0000 01xx xxxx xxxx xxxx xxxx xxxx	0x00400000	4M	QuadSPI Flash
01xx xxxx xxxx xxxx xxxx xxxx xxxx	0x04000000	64M	DDR3 SDRAM

Table 5.1: Address Regions

5.1 Peripheral I/O Control

Tbl. 5.3 shows the addresses of various I/O peripherals included as part of the SoC. We'll walk through each of these peripherals in turn, describing how they work.

5.1.1 Interrupt Controller

Currently defined bus interrupts are listed in Tbl. 5.4.

Base	Size(W)	Purpose
0x0c0000000	1	Primary Zip PIC
0x0c0000001	1	Watchdog Timer
0x0c0000002	1	Bus Watchdog Timer
0x0c0000003	1	Alternate Zip PIC
0x0c0000004	1	ZipTimer-A
0x0c0000005	1	ZipTimer-B
0x0c0000006	1	ZipTimer-C
0x0c0000007	1	ZipJiffies
0x0c0000008	1	Master task counter
0x0c0000009	1	Master prefetch stall counter
0x0c000000a	1	Master memory stall counter
0x0c000000b	1	Master instruction counter
0x0c000000c	1	User task counter
0x0c000000d	1	User prefetch stall counter
0x0c000000e	1	User memory stall counter
0x0c000000f	1	User instruction counter
0x0c0000010	1	DMA command register
0x0c0000011	1	DMA length
0x0c0000012	1	DMA source address
0x0c0000013	1	DMA destination address
0x0c0000040	1	MMU context register
0x0c0000080	32	MMU TLB

Table 5.2: ZipSystem Addresses

Name	Address	Width	Access	Description
VERSION	0x0100	32	R	Build date
PIC	0x0101	32	R/W	Bus Interrupt Controller
BUSERR	0x0102	32	R	Last Bus Error Address
PWRCOUNT	0x0103	32	R	Ticks since startup
BTNSW	0x0104	32	R/W	Button/Switch controller
LEDCTRL	0x0105	32	R/W	LED Controller
GPIO	0x0106	32	R/W	GPIO controller
GPS-SETUP	0x0107	29	R/W	GPS UART Setup register
CLR-LEDx	0x0108	32	R/W	Color LED controller

Table 5.3: I/O Peripheral Registers

Name	Bit Mask	Description
INT_BUTTON	0x0001	A Button has been pressed.
INT_SWITCH	0x0002	The Scope has completed its collection
INT_PPS	0x0004	Top of the second
INT_RTC	0x0008	An alarm or timer has taken place (assuming the RTC is installed, and includes both alarm or timer)
INT_NETRX	0x0010	A packet has been received via the network
INT_NETTX	0x0020	The network controller is idle, having sent its last packet
INT_UARTRX	0x0040	A character has been received via the UART
INT_UARTTX	0x0080	The transmit UART is idle, and ready for its next character.
INT_GPIO	0x0100	The GPIO input lines have changed values.
INT_FLASH	0x0200	The flash device has finished either its erase or write cycle, and is ready for its next command. (Alternate config only.)
INT_SCOPE	0x0400	A scope has completed collecting.
INT_GPSRX	0x0800	A character has been received via GPS
INT_SDCARD	0x1000	The SD-Card controller has become idle

Table 5.4: Bus Interrupts

5.1.2 Last Bus Error Address

5.1.3 General Purpose I/O

5.1.4 UART Data Register

5.2 Debugging Scopes

5.3 Internal Configuration Access Port

5.4 Real–Time Clock

5.5 On-Chip Block RAM

5.6 Flash Memory

6.

Clocks

Name	Source	Rates (MHz)		Description
		Max	Min	
i_clk_100mhz	Ext	100 MHz		100 MHz Crystal Oscillator
s_clk	PLL	200 MHz		Internal Logic, Wishbone Clock
ram_clk	PLL	200 MHz		DDR3 SDRAM Clock
o_sck	Logic	108 MHz	50 MHz	QSPI Flash clock
o_sdclk	Logic	50 MHz	100 kHz	SD–Card clock

Table 6.1: OpenArty clocks

7.

I/O Ports

Table. 7.1 lists the various I/O ports associated with OpenArty.

Port	Width	Direction	Description
i_clk_100mhz	1	Input	Clock
o_qspi_cs_n	1	Output	Quad SPI Flash chip select
o_qspi_sck	1	Output	Quad SPI Flash clock
io_qspi_dat	4	Input/Output	Four-wire SPI flash data bus
i_btn	2	Input	Inputs from the two on-board push-buttons
o_led	4	Output	Outputs controlling the four on-board LED's
o_pwm	1	Output	Audio output, via pulse width modulator
o_pwm_shutdown_n, 1		Output	Audio output shutdown control
o_pwm_gain	1	Output	Audio output 20 dB gain enable
i_uart	1	Input	UART receive input
o_uart	1	Output	UART transmit output
o_uart_cts	1	Output	H/W flow control response, true if the internal single-byte receive buffer is empty.
i_uart_rts	1	Input	H/W flow control, true if the PModUSBUART wishes to send a byte
i_kp_row	4	Output	Four wires to activate the four rows of the keypad
o_kp_col	4	Output	Return four wires, from the keypads columns
i_gpio	14	Output	General purpose logic input lines
o_gpio	14	Output	General purpose logic output lines
io_scl	1	Input/Output	I2C clock port
io_sda	1	Input/Output	I2C data port

Table 7.1: List of IO ports