AEMB 32-bit Microprocessor Core Datasheet

Shawn Tan Ser Ngiap
shawn.tan@aeste.net

November 11, 2007

Contents

1 Introduction

The AEMB is a clean room implementation of the EDK3.2 software compatible Microblaze core using information from the Internet. It is cycle and instruction compatible to the MB for most software commands. It is not meant as a drop in replacement for the Microblaze as it is not architecturally compatible. It uses the WISHBONE bus instead of LMB/OPB.

This is a CPU core that is capable of moving and manipulating data to and from memory. It does not have any peripherals nor interrupt controllers although support for external interrupts is provided. Any peripherals and their respective registers could be mapped to the data memory space. It has a separate instruction, data and FSL bus.

This core is fairly similar to the original Microblaze from the point of view of software. Therefore, this list of features are mainly for the benefit of people unfamiliar with the original Some of the main features are:

1. Harvard architecture with a separate 32-bit instruction and data busses. The address space for each bus can be separately configured with core parameters.
2. Pipelined operation with a 3-stage integer pipeline. The pipeline is capable of executing one instruction per clock. This short pipeline allows it to context switch quickly.
3. Support for hardware multiplier and barrel shifter. Implementation of a single cycle multiplier and barrel shifter will improve software performance.
4. Support for GET/PUT instructions. The GET/PUT instructions are implemented as a seperate FSL bus. Other peripherals or FIFOs can be connected to this bus.
5. Small core size with an excellent performance. It has an equivalent gate count of about 38k gates at 136 MHz in a Xilinx Virtex4 FPGA (without multiplier).
6. Mature software development toolchain as it is software compatible with the original. Operating system support for the original core includes uClinux/FreeRTOS.

2 External Signals

The core uses a WISHBONE compatible bus for it's on-chip bus. It is cycle compatible with WISHBONE classic cycles. This allows the core to be quickly integrated with many other devices that use this bus. It supports the main legacy bus signals. In addition to this, it also uses some other signals that are described below.

2.1 Ports

The core ports are broken into four groups: Instruction Bus, Data Bus, FSL Bus and System signals. They are all listed in table 2.1 with short descriptions. All the signals are active high as per WISHBONE documentation.

The signal functions should all be quite self explanatory. For detailed cycle arbitration and timings, please refer to the official WISHBONE specifications. Some of the WISHBONE signals are missing as they may not be necessary like the write-enable signal for the instruction bus.

NAME	SIZE	I/O	DESCRIPTION
IWB_ADR_O	30	Out	Instruction bus address
IWB_STB_O	1	Out	Instruction bus request/strobe
IWB_DAT_I	32	In	Instruction bus data word
IWB_ACK_I	1	In	Instruction bus acknowledge
DWB_ADR_O	30	Out	Data bus address
DWB_DAT_O	32	Out	Data bus data write word
DWB_STB_O	1	Out	Data bus request/strobe
DWB_WRE_O	1	Out	Data bus write/read enable
DWB_SEL_O	4	Out	Data bus byte lane select
DWB_DAT_I	32	In	Data bus data read word
DWB_ACK_I	1	In	Data bus acknowledge
FSL_ADR_O	13	Out	FSL bus address
FSL_DAT_O	32	Out	FSL bus data write word
FSL_STB_O	1	Out	FSL bus request/strobe
FSL_WRE_O	1	Out	FSL bus write/read enable
FSL_DAT_I	32	In	FSL bus data read word
FSL_ACK_I	1	In	FSL bus acknowledge
SYS_INT_I	1	In	Positive edge triggered interrupt
SYS_CLK_I	1	In	Master clock signal
SYS_RST_I	1	In	Master active low reset

Table 2.1: External signal names and descriptions.

3 Architecture Blocks

The top-level core (CORE) has a separate instruction and data memory bus without cache memory. This can be used as part of an SoC with internal devices attached to the data memory bus. It is also the main top-level core as it contains all the functional sub-blocks. There are two top-level cores supplied.

3.1 Top Level Cores

All designs should use the EDK32 top level core. The old CORE top level is only available for compatibility reasons. It will be eventually phased off once the EDK32 top level core matures. The CORE top level will no longer be supported.

3.2 Core Parameters

The EDK32 core includes an optional hardware multiplier and barrel shifter. The optional hardware components are configured using core parameters. The width of both the instruction and data bus can be specified in the core parameters.

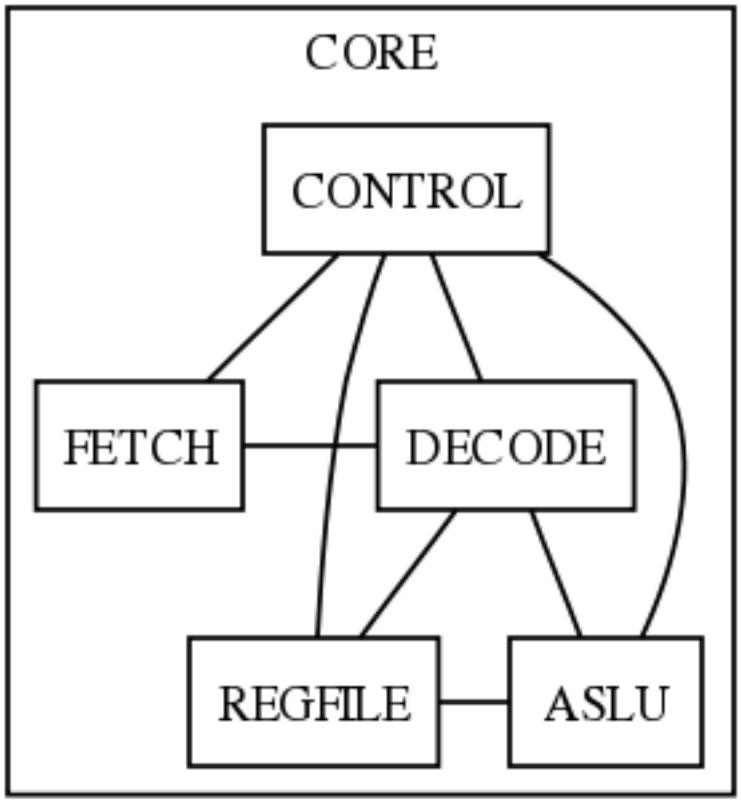

Figure 3.1: AEMB block hierarchy and links

4 Functional Operation

Although functionally similar to the Microblaze, the AEMB is not a drop-in substition. The AEMB can be used as an alternative core that can execute compatible code. This chapter documents some of the operation details of the core.

4.1 Reset Operation

The master reset signal for the core is SYS_RST_I. During reset, all the internal registers for the core are set to their reset values. All interrupts are disabled by the MSR_IE register value. After reset, the processor begins to fetch instructions from memory location 0x00000000.

4.2 Pipeline Operation

The integer pipeline for the AEMB is a single issue and in-order execution pipeline. It consists of three main stages: *FETCH & DECODE*, *LOAD & EXECUTE* and *BRANCH & STORE*. Each stage is synchronised onto the positive edge SYS_CLK_I signal.

Stalls The pipeline is stalled for *any* incomplete memory operation. This means that, unless the memory is fast enough, the processor will stall the entire pipeline. This includes both instruction and data memory access. During a branch it is possible for the AEMB to use a branch delay slot.

4.3 Interrupt Operation

The core has support for positive edge triggered interrupt on the SYS_INT_I signal. The interrupt latency is between 3-5 cycles. Interrupts will not trigger between non-atomic instructions such as IMMI.

5 Software & Simulation

The AEMB core is capable of executing C code compiled with the Microblaze GCC toolchain. Development of the core was simulated using GPLCVER 2.11a and Icarus Verilog 0.8.5 simulators. Simulatio code was compiled with GCC 3.4.1 (Xilinx EDK 8.1.01 Build EDK_I.19.4 061107). However, since the core does not implement certain instructions, certain compilation flags should be used.

```
$ mb-gcc -g -mxl-soft-div -msoft-float
```

Some of the hardware components are parameterisable. If a hardware multiplier and barrel shifter is available, other compilation flags can be used to speed up software performance.

```
$ mb-gcc -g -mno-xl-soft-mul -mxl-barrel-shift
```

An example compilation script is provided in the */sw/gccrom* shell script. This script takes any optional GCC arguments including the C filename. It will generate a suitable simulation ROM and place it in the */sim/aeMB.rom* file. The */sw/c/aeMB_testbench.c* file shows an example C algorithm to calculate Fibonnaci numbers.

```
$ ./gccrom c/aeMB_testbench.c
```

The */sim/verilog/testbench.v* verilog file is an example simulation testbench. This simulation will load and run the software that is located in the rom file. It is highly tailored to run tests based on the example testbench C code. The */sim/cversim* and */sim/iversim* are scripts to run the simulation with either CVER or Icarus verilog. These scripts take the programme arguments as well as the simulation testbench file to use.

```
$ ./cversim verilog/edk32.v
```

! → All these files are provided as examples and should be used as a baseline. For user applications, custom C software and testbench code should be written, which can be based on these files and scripts.

6 Implementation

! → These non-constrained synthesis results from Xilinx ISE v9.1i are for the core. These are not a benchmark but are useful as an estimate of chip resources and performance.

The core has been envisioned to be used as part of a larger SoC. Hence, it has been designed with a small size as an objective.

It is possible to further optimise the design as the critical path runs through the main ALU. About 44% of this critical path is due to routing, not logic.

```
Device utilization summary:
---------------------------

Selected Device : 4vlx25ff668-12

 Number of Slices:                     1268  out of  10752    11%
 Number of Slice Flip Flops:            272  out of  21504     1%
 Number of 4 input LUTs:               2082  out of  21504     9%
    Number used as logic:              1698
    Number used as RAMs:                384
 Number of IOs:                         248
 Number of bonded IOBs:                 248  out of    448    55%
 Number of GCLKs:                         1  out of     32     3%

Timing Summary:
---------------
Speed Grade: -12

   Minimum period: 7.315ns (Maximum Frequency: 136.710MHz)
   Minimum input arrival time before clock: 7.404ns
   Maximum output required time after clock: 4.005ns
   Maximum combinational path delay: No path found
```

---

7 License

GNU LESSER GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. ¡http://fsf.org/¿ Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.

d) Do one of the following:

0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.