

EFM[®]32

... the world's most energy friendly microcontrollers

External Bus Interface

AN0034 - Application Note

A decorative graphic on the right side of the page consisting of concentric circles. The outermost ring is light green, followed by a yellow-green ring, a blue ring, and a dark blue inner circle. The numbers 0, 1, 2, 3, and 4 are arranged in a horizontal line across the center of these circles, with 0 in the light green ring, 1 in the yellow-green ring, 2 in the blue ring, 3 in the dark blue ring, and 4 in the center of the dark blue circle.

Introduction

This application note shows how to use the EBI module in the EFM32 and access an external SRAM on board the DK

This application note includes:

- This PDF document
- Source files (zip)
 - Example C-code
 - Multiple IDE projects

1 Introduction

Until the advent of modern high speed serial buses, parallel buses have been historically extensively used in computers, for example the ISA, ATA, SCSI, PCI, Front side bus or the IEEE-1284 / Centronics printer port. The main reason for this usage was that a n-bit parallel channel will transmit the data n times faster than a serial channel, assuming the same clock speed. However, a parallel channel will generally have additional control signals such as a clock, indicating that the data is valid, and possibly other signals for handshaking and directional control of data transmission. Generally the criteria of choice include:

- Speed: in spite of the theoretically n times higher speed, in a parallel bus the clock skew reduces the speed of every link to the slowest of all of the links.
- Cable length: Crosstalk creates interference between the parallel lines and the maximum length of a parallel data link will be shorter than that of a serial link
- Ease of implementation: Parallel data links are easily implemented in hardware and firmware, while serial links require more processing

With the apparition of high speed serial buses, leading to replacements which are nowadays industry standards (such as USB printer port for IEEE 1284, SATA for parallel ATA, or Firewire for SCSI), the use of parallel buses has been restrained to very high throughput situations, such as a processor-to-memory bus, or high quality digital video equipment data.

2 The EFM32 EBI

The External Bus Interface present on EFM32 microcontrollers is a versatile parallel address/data bus that provides access to common external parallel interface devices such as SRAM, FLASH, ADCs and LCDs. The interface is memory mapped into the address bus of the Cortex-M3, which enables seamless software access without the need for IO-level access each time a read or write is performed.

Since the devices appear as a part of the EFM32s internal memory map, they are extremely simple to use. When the processor performs read or writes to the address range of the EBI, the EBI handles data transfer to and from the external device.

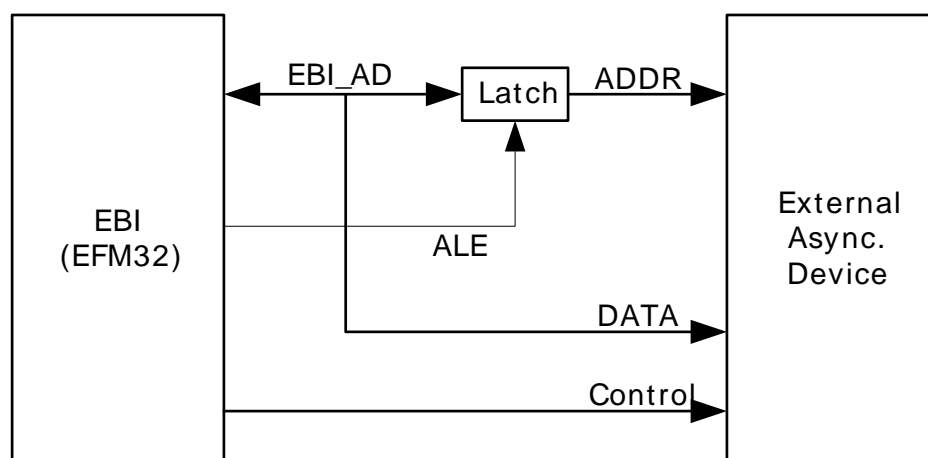
The EBI is available in Energy Mode 0 and Energy Mode 1 and may be interfaced by the DMA, thus enabling for example DMA operation of an external memory chip in EM1.

The data and address lines are multiplexed in order to reduce the number of pins required to interface the external devices. The timing is adjustable to meet specifications of the external devices. The interface is limited to asynchronous devices (no clock signal is available)

The EBI has 16 pin connections, denoted EBI_AD00 through EBI_ADD15, which can be used for both address and data lines. In 8-bit address mode, the address and data both fit into these pins. If more address bits or data bits are needed, external latches can be used to support up to 24-bit addresses or 16-bit data. The following 3 modes are available:

1. 8-bit address and 8-bit data. In this mode, EBI_AD[07:00] are used as data pins and EBI_AD[15:08] are the address pins. No external circuitry and no multiplexing is required
2. 16-bit address and 16-bit data. In this mode, EBI_AD[15:00] are used in a multiplexed fashion, first as address bits, then as data bits. Multiplexing is achieved by the usage of the EBI_ALE, EBI_CS_n and EBI_RE_n signals with external latches (Figure 2.1 (p. 3)), as per the reference manual for the EFM32 device.
3. 24-bit address and 8-bit data. In this mode, the bus is used also in a multiplexed fashion, in a two steps process. In the first step, EBI_AD[15:08] are used to pass address bits ADDR[23:16] and simultaneously EBI_AD[07:00] are used to pass address bits ADDR[15:08]. In the second step, EBI_AD[15:08] are used to pass address bits ADDR[07:00] and simultaneously EBI_AD[07:00] are used to pass the 8-bit data, DATA[07:00]. Multiplexing is also achieved by the usage of the EBI_ALE, EBI_CS_n and EBI_RE_n signals with external latches. For more details, please consult the reference manual for the EFM32 device.

Figure 2.1. EBI Address Latch Setup



3 Gecko DK hardware fixture

The EFM32 Gecko development kits include a single chip 4 Mbit parallel bus SRAM. This is a 16-bit wide datapath memory (as such it is organized as 256K words of 16 bits each). The exact part number on the current DVKs is CY62147EV30LL-45BVXI from Cypress and the chip has a 48-ball FBGA package. As a memory with 256K addressable locations, the CY62147EV30 will support 18 bit addresses, since $256K = 2^{18}$. Note with respect to the DVK schematic that the chip is represented as having actually 21 address pins, named A0 through A20, but the uppermost three A[20:18] are not connected internally on the CY62147EV30. (The DVK schematic / PCB layout as given is useful for connecting SRAMs from the same family, but up to 8 times larger). As such, the address space is determined by the A[17:0] pins.

The EFM32G290 MCU board (without an LCD) is recommended to be used for this application, in conjunction with the EFM32-G2xx-DK. (Because of the presence of the LCD on the EFM32G890 MCU board, this board with its corresponding DVK exhibit limited EBI support.)

On the Gecko DVK, the role of the "latch" is played by an FPGA known as a "board controller" (Xilinx Spartan XC3S200A series). The SRAM can be accessed through the board controller and software support is available to drive the board controller in order to access the SRAM via EBI (and disable SPI) in either 8-bit or 16-bit data width mode. The DVK firmware version must be 1v6p0b16 or higher, in order for the example to work. Also, the AEM STATE must be set to EFM. This state can be toggled by pushing the AEM button on the DVK. The status is indicated in the top right hand corner of the TFT display.

4 EFM32 Implementation

The application note firmware uses the EFM32 EBI in the 16-bit address/16-bit data mode, in order to match the native format of the Cypress SRAM memory. It implements memory read and write operations as translations from the external SRAM to the internal SRAM and vice-versa of a data chunk specified by start and end pointers. The below considerations apply with respect to the EFM32G290 MCU.

In the chosen implementation mode the 16-bit address is organized in 2-byte chunks at memory addresses aligned to 2-byte offsets. Consequently, the LSB of the 16-bit address will always be 0. In order to double the address space, the 16-bit address is internally shifted one bit to the right so that the LSB of the address driven into the EBI_AD bus, i.e. the EBI_AD[0]-bit, corresponds to the second least significant bit of the address, i.e. ADDR[1]. At the external device, the LSB of the address must be tied either low or high in order to create a full address

Now the internal MCU SRAM data space is a space determined by 32-bit location addresses. At each address, of course, a 32-bit data word is located. The addresses of this space are between 0x2000000 and 0x20003FFF, thus a total of decimal 16384 or 16K locations of 32-bit SRAM words available.

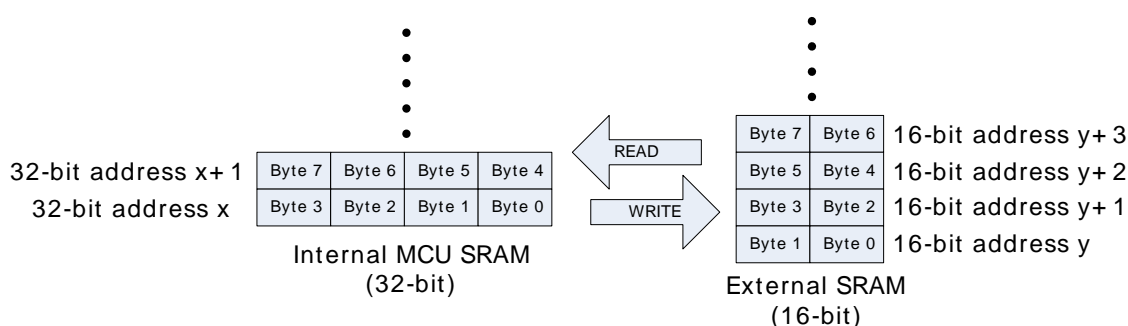
The EBI is mapped at higher memory addresses. There are 4 EBI regions, each corresponding to a physical chip select pin on the MCU, to facilitate memory banking. Each EBI region has a size of decimal 67108864 or 64M locations of 32-bit words each as follows:

- Region 0 selected by EBI_CS0 : from 0x80000000 to 0x83FFFFFF
- Region 1 selected by EBI_CS1 : from 0x84000000 to 0x87FFFFFF
- Region 2 selected by EBI_CS2 : from 0x88000000 to 0x8BFFFFFF
- Region 3 selected by EBI_CS3 : from 0x8C000000 to 0x8FFFFFFF

The SRAM memory in this example is mapped starting at address 0x84000000 (the memory is being selected by pulling the EBI_CS1 pin low).

Writing to the external RAM appears to the programmer as simply moving 32-bit data between the 0x2000000 - 0x20003FFF address space (internal SRAM) and the 0x84000000 - 0x87FFFFFFF address space (mapping of the external SRAM) and reading is vice-versa. The physical effect on the actual external SRAM chip, in the sense of the order of bytes being read/written is described by the Figure 4.1 (p. 5) .

Figure 4.1. Read/Write



Note that since only 16 address bits are used for the external SRAM (while its maximum capability is determined by 18 address bits), the two most significant address bits of the external bus will always be zero and reading/writing will occur in this example only to/from the lower quarter.

The parameters of the write function are as follows:

- writeAddr - the (16-bit external SRAM address) of the location where the first element of the buffer is to be written
- data - represents the actual data that is written at the address indicated by the writeAddr

- bufferSize - represents the length of the data buffer (number of 16-bit words in the buffer). In the write function, a for loop is used that manipulates the data until the end of the buffer is reached. After each write operation, where the DVK_EBI_writeRegister is used, the address is incremented to the next one.

The parameters of the read function are as follows:

- readAddr - the (16-bit external SRAM address) address of the location where the reading of the first element starts
- bufferSize - represents the length of the data buffer

After the write and read processes are finished, the answer buffer is tested against the initial buffer to compare. If everything is OK, an on-board LED will stay turned on. If not, the LED will be toggled signaling that the process failed. The algorithm can be (and has been) tested at 32MHz clock speed, changing write timings, read timings and address timings until the process fails. (The corresponding failed register value is indicated in a code example.)

Note

To build the project correctly the symbol DVK_EBI_CONTROL must be defined in the preprocessor project configurations

Note

It is recommended to use a pull-up in the CSn lines to have these pins on a defined state

5 Revision History

5.1 Revision 1.04

2012-04-20

Adapted software projects to new peripheral library naming and CMSIS_V3.

5.2 Revision 1.03

2012-03-14

Fixed compilation error in CodeSourcery projects.

5.3 Revision 1.02

2011-10-21

Updated IDE project paths with new kits directory.

5.4 Revision 1.01

2011-07-18

Fixed errors in main.c and added line to wait for AEM state.

Added note in document that AEM state must be waited for.

5.5 Revision 1.00

2011-03-28

Initial revision.

A Disclaimer and Trademarks

A.1 Disclaimer

Energy Micro AS intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Energy Micro products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Energy Micro reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Energy Micro shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products must not be used within any Life Support System without the specific written consent of Energy Micro. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Energy Micro products are generally not intended for military applications. Energy Micro products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

A.2 Trademark Information

Energy Micro, EFM32, EFR, logo and combinations thereof, and others are the registered trademarks or trademarks of Energy Micro AS. ARM, CORTEX, THUMB are the registered trademarks of ARM Limited. Other terms and product names may be trademarks of others.

B Contact Information

B.1 Energy Micro Corporate Headquarters

Postal Address	Visitor Address	Technical Support
Energy Micro AS P.O. Box 4633 Nydalen N-0405 Oslo NORWAY	Energy Micro AS Sandakerveien 118 N-0484 Oslo NORWAY	support.energymicro.com Phone: +47 40 10 03 01

www.energymicro.com

Phone: +47 23 00 98 00

Fax: + 47 23 00 98 01

B.2 Global Contacts

Visit **www.energymicro.com** for information on global distributors and representatives or contact **sales@energymicro.com** for additional information.

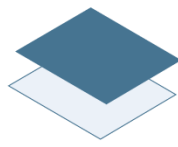
Americas	Europe, Middle East and Africa	Asia and Pacific
www.energymicro.com/americas	www.energymicro.com/emea	www.energymicro.com/asia

Table of Contents

1. Introduction	2
2. The EFM32 EBI	3
3. Gecko DK hardware fixture	4
4. EFM32 Implementation	5
5. Revision History	7
5.1. Revision 1.04	7
5.2. Revision 1.03	7
5.3. Revision 1.02	7
5.4. Revision 1.01	7
5.5. Revision 1.00	7
A. Disclaimer and Trademarks	8
A.1. Disclaimer	8
A.2. Trademark Information	8
B. Contact Information	9
B.1. Energy Micro Corporate Headquarters	9
B.2. Global Contacts	9

List of Figures

2.1. EBI Address Latch Setup	3
4.1. Read/Write	5



ENERGY[®]
micro

*Energy Micro AS
Sandakerveien 118
P.O. Box 4633 Nydalen
N-0405 Oslo
Norway*

www.energymicro.com