Engineering Bulletin

AN2297/D
Rev. 0.1, 6/2002

Implementing an MGT5100 Ethernet Driver

MOTOROLA intelligence everywhere™ | digital dna™

General Description

To write an ethernet driver for the MGT5100 Faster Ethernet Controller (FEC) under CodeWarrior EPPC6.1, follow this guide. This document describes simple FEC register initialization and buffer descriptor (BD) handling and SmartComm (MGT5100 DMA engine). Buffer descriptors are software data structures documented in the MGT5100 User's Guide used to send and receive ethernet frames from the FEC. This document is intended to aid developers implementing their own MGT5100 ethernet driver. A simple example test program is also provided to show a single transmission and reception of a data packet.

Tranceivers

The FEC requires an external ethernet tranceiver chip (PHY) to connect to a physical interface. There are two basic types of PHY: those with a seven-wire interface and the standard 18-wire media independent interface (MII). The code is written for an Intel LXT905LE seven-wire PHY and an Intel LXT971ALE MII PHY. Other seven- and 18-wire PHYs may be similar to these such that one may use the same initialization code.

Provided Files

Ethernet Driver

mgt5100fec.c

This file provides register initialization for the MGT5100 FEC as well as data structure initialization for the FEC buffer descriptor handling code. It also provides primitive runtime send and receive frame functions and their helpers, which can be called from the user's kernel services and interrupt handlers.

eth_task.dist.s

Assembler-ready SmartComm opcodes and registers are encapsulated in this file. These are the backend of the MGT5100_Eth_Send() and MGT5100_Eth_Receive() functions defined in mgt5100fec.c that manage buffer descriptors and the FEC's transmit and receive FIFOs. The directives used are that of GNU assembler or CodeWarrior assembler depending on if __MWERKS__ is defined at the top of the file. It is possible the directives will need to be modified when a different assembler is used to compile this file.

loadtask.c

The loadtask() function loads the ethernet task from eth_task.dist.s into the SmartComm module.

nif.c

This module implements a sample network interface (nif) for integration between the simple handlers in mgt5100fec.c and higher level network protocol stacks. The user may wish to reimplement this to better match the interface between their own network code and interrupt handlers.

mgt5100fec.h

This file contains FEC register definitions, software data types and declarations.

cpu/ppc/mgt5100/mgt5100.h

This file contains MGT5100-specific PowerPC definitions. In particular it contains machine base address register (MBAR) offsets for the GPIO, SmartDMA, ethernet and SRAM peripherals.

cpu/ppc/mgt5100/sdma.h

This file contains SmartDMA definitions and declarations, interrupt and task enable/disable macros and buffer descriptor parameter addresses.

cpu/ppc/mgt5100/int_ctrl.h

This file contains MGT5100 interrupt controller register definitions.

cpu/ppc/mgt5100/gpio_std.h

This file contains MGT5100 general purpose I/O (GPIO) register definitions.

board.h

This file contains board-specific definitions, notably the MBAR address.

types.h

This file contains definitions for machine word sizes and general programming.

Example Program

main.c

This file sets up the environment for testing and then calls the Ethernet test routine

interrupt.c

This file contains an interrupt handler to trap all exceptions defined in the exception table.

test_ethernet.c

This file contains code to initialize FEC and SmartComm, and to execute the test in the example program.

CodeWarrior Files

MGT5100_ETHERNET_EXAMPLE_data/

This directory contains CodeWarrior project files needed to compile and run the example program.

eppc_exceptions.asm

This file contains the exception vector table for MGT5100.

5100_Mot_EVB_RAM.lcf

This file contains linker commands that CodeWarrior uses to define memory regions and other link-time parameters.

5100_Mot_EVB_init.c

This file contains routines to intialize the IceCube board.

Prefix_ethernet.h

This file contains Ethernet example definitions.

Prefix_5100_Mot_EVB_ISR.h

This file contains board Ethernet example environment setup definitions.

5100_Mot_EVB_init.cfg

This file contains CodeWarrior target initialization for MGT5100.

User Defined Functions

The follwing functions are required by but not defined in the provided driver code. They are provided in the example program.

NBUF *nbuf_allocate(int size)

This function allocates a network buffer descriptor. It is a specific case of an ANSI malloc() library call. The size argument is the requested buffer size in bytes.

void nbuf_release(NBUF *frame)

The complement of nbuf_allocate(), this function releases a network buffer descriptor back to the free pool much like the ANSI free() library call.

phy_type_t board_eth_phy_type(void)

This function provides runtime configuration of a board's ethernet transceiver type defined by the phy_type_t enum in mgt5100fec.h.

phy_model_t board_eth_phy_model(void)

This function provides runtime configuration of a board's specific ethernet tranceiver model name defined by the phy_model_t enum in mgt5100fec.h.

void board_msecond_wait(int mseconds)

This function delays execution for mseconds milliseconds.

int printf(char *format, ...)

This is the C stdio printf(3) function. It is used for debugging purposes only and required if the ETH_DEBUG macro is defined.

Order of Operations

1. Set MBAR to 0xf0000000 in CodeWarrior's target initialization file.
2. Initializes SDRAM controller by CodeWarrior's target initialization file.
3. Call the test_ethernet() function to begin the following Ethernet test operations.
4. Set the element taskBAR address in sdma_regs to MBAR_SRAM to point to the base of the Task Table.
5. Call the loadtask() function to load receive and transmit Ethernet SDMA tasks into the SRAM.
6. Call the MGT5100_Eth_Init() function to initialize software data structures.
7. Call the nif_bind_proto() function to register a receive callback function with a given protocol.
8. Call the MGT5100_Eth_Start() function to configure the FEC control registers, configure the MII PHY registers (if present, 7-wire PHYs need no initialization) and start the SmartComm task.
9. Call the MGT5100_Eth_Send() function to send an Ethernet data frame from a source address to a destination address. In the case of the test example included, source and destination have the same address.
10. Call the MGT5100_Eth_Receive() function in the interrupt handler to manage received frames. However, in this test example, test_rx_buf() function is called instead to check the status of the receive buffer.
11. Call the MGT5100_Eth_Stop() function to stop sending and receiving ethernet frames. This causes the SmartComm task and the FEC to be disabled.

Ethernet Test Example

Follow these steps in order to run the Ethernet Test Example using CodeWarrior EPPC6.1 in conjunction with Abatron BDI 2000.

1. Install and set up CodeWarrior EPPC6.1.
2. Install and set up Abatron BDI2000.
3. Set up MGT5100 IceCube board.
4. Power up Abatron.
5. Power up IceCube board.
6. Make a serial connection (57600, 8N1) to the IceCube board to view standard output from the execution of the test program.
7. Open up the Ethernet Example Test CodeWarrior Project “MGT5100 ETHERNET EXAMPLE.mcp.”
8. Click “Run” Button from the CodeWarrior Stationary or select “Run” from the Project Menu.
9. The output of the test program should closely resemble the following sample output.

Test Program Sample Output

```
Welcome to CodeWarrior EPPC 6.1! 2 RAM Version Jun  7 2002 01:25:28
Welcome to CodeWarrior EPPC 6.1! 1 RAM Version Jun  7 2002 01:25:28

**********************************

Ethernet Test Settings
ETH_DEBUG: 0x1ff
Promiscous mode used
Loop mode used
Receive interrupt disabled
Transceive interrupt disabled

base task number = 0, number of tasks = 2
task_org = 0x00007f00
start1 = 0x00000009, start2 = 0xf00041b8
TDT start = 0x00000040, end = 0x000001b8
MGT5100_Eth_Start is called
Mii reg 0: 0x1000
Mii reg 1: 0x782d
Mii reg 2: 0x0013
Mii reg 3: 0x78e2
Mii reg 4: 0x01e1
Mii reg 5: 0x45e1
Mii reg 6: 0x0007
Mii reg 7: 0x2001
Mii reg 8: 0x0000
Mii reg 16: 0x0084
Mii reg 17: 0x4780
Mii reg 18: 0x0000
Mii reg 19: 0x00f4
Mii reg 20: 0x0422
SDMA task will be enabled now
```

```
MGT5100_Eth_Send is called
Frame is sent
RX Buffer 0 not empty
   Status: 0x0800
   Data Length: 518
MGT5100_Eth_Stop is called
Mii reg 0: 0x1000
Mii reg 1: 0x782d
Mii reg 2: 0x0013
Mii reg 3: 0x78e2
Mii reg 4: 0x01e1
Mii reg 5: 0x45e1
Mii reg 6: 0x0007
Mii reg 7: 0x2001
Mii reg 8: 0x0000
Mii reg 16: 0x0084
Mii reg 17: 0x4780
Mii reg 18: 0x0000
Mii reg 19: 0x0000
Mii reg 20: 0x0422
```

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140 or 1-800-441-2447

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.;
Silicon Harbour Centre, 2 Dai King Street,
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

1-800-521-6274

HOME PAGE:

http://www.motorola.com/semiconductors