Memory-to-Memory Transfers Using the SA-1110 DMA

Application Note

September 2000

Order Number: 278353

Application Note

Overview

The SA-1110 processor has two on-board DMA controllers; one of which is dedicated for use with the LCD controller; the other consists of six independent DMA channels, which can be configured to interface to any serial port (memory-to-I/O or I/O-to-memory). Two DMA channels are required to support full duplex to any one of the serial I/O ports. Because the SA-1110 is not equipped with a memory-to-memory DMA controller, it is not easily possible to transfer from one memory location to another.

This Application Note responds to questions about transferring data from memory to memory on the SA-1110 processor by connecting two DMA channels to a serial port configured in loopback mode to allow memory-to-memory transfer in a non- “fly-by” mode.

The SA-1110 processor can be configured to move blocks of memory from one location to another, and to move a block of memory for one location to a signal address or a signal address to a block of similar memory. The example described in this Application Note explains how to configure the SA-1110 DMAs to transfer data from one block of memory to another. This configuration is simple to verify.

This application note should be used in conjunction with the *Intel® StrongARM™ SA-1110 Microprocessor Advanced Developers Manual*, Order Number 278240-002.

Concept

For operations that do not need to transfer blocks larger than 8KB of memory, the concept is simple.

1. First initialize the HSSP port for 4Mb/s and loopback mode.
2. Next, enable the HSSP port. The HSSP port was selected just as an example serial port capable of loopback mode. The decision on which port to use should be based on two criteria: (1) available serial ports that support loopback mode in full duplex, and (2) the throughput required for the application.
3. Once the serial port has been configured for loopback mode, two DMA channels must be configured and enabled: one for the receive channel, and the other for the transmit channel. The on-board HSSP port will interrupt once the receive buffer detects the End-of-Frame status.
4. To complete the transfer, the trailing bytes must be cleared from the HSSP receive buffer.

Performance

Tests determined that the effects of the CPU core accessing the memory bus were minimal during a DMA memory-to-memory transfer. Data throughput tests determined the transfer rate using the HSSP port in loopback mode averaged approximately 4 Mb/s. During the test, data was driven continuously to an external register requiring external bus access while transferring data from memory-to-memory using the DMA controller.

Understand that the only DMA transfers that existed during the test were the two DMA channels that performed the memory-to-memory transfer. Any additional DMA transfers with a higher priority than those assigned to perform the memory-to-memory transfer may reduce the 4 Mb/s transfer rate. Designers may need to run additional tests to determine the true throughput with additional higher priority DMA transfers.

Most designers will choose not to use two DMA channels to transfer data from memory to memory because data can be moved between memory locations much faster using the CPU core. Others may choose to use this technique to minimize the load on the CPU core and take advantage of “free resources” that are not being used on the SA-1110 processor in their original design.

Details

The example in this Application Note provides details on how to program the DMA controller and the HSSP port to transfer data memory to memory at a rate of 4Mb/s for any size transfer. In addition, information is provided on which DMA channels to select.

To begin, the mem_to_mem() procedure should accept the address of the source memory, destination memory, and the number of bytes to be transferred.

It is a good idea to have an integer value returned, which determines the success of the initiation of the transfer. For example, a return value of zero would indicate a successful initiation of the transfer. A return value of 1 would indicate a DMA currently in use, and a return value of 2 would indicate values passed to the procedure are invalid.

The first task required by the mem_to_mem() procedure will be to assign the parameters passed in to global variable to be accessible by the interrupt routine. In addition to those passed in, a global variable should be present to indicate if the DMA mem_to_mem() is in use or complete. The interrupt routine clears the DMA-in-use global variable only if the calculated data remaining to transfer is zero.

To initiate the transfer, the procedure would first initialize the IRDA port for 4Mb/s and loopback mode. Next, the DMA receive channel should be initialized and enabled to interact with the HSSP receive buffer. The value of the count register must be determined before the receive and transmit DMA channels can be initialized. Once the DMA receive channel has been initialized, it is safe to initialize and enable the DMA transmit channel to begin transfer, and pass control back to the calling program.

Be careful when determining which DMA channel to use for the receive and transmit buffers of the HSSP port. It is recommended that the receive DMA channel should be assigned a higher priority than the transmit channel as this will prevent the HSSP receive logic from generating an overrun error. The channels are serviced in a fixed priority, with channel 0 being the highest and channel 5 being the lowest.

Do not enable either the DMA transmit or receive channel interrupt. After the transfer of the first data block, the HSSP port will detect an end-of-frame (EOF) status, and generate an interrupt. The HSSP port interrupt routine clears the trailing bytes from the buffer and recalculates the number of bytes remaining to transfer. If there are none, the interrupt routine should clear the DMA-in-use global variable, indicating the transfer is done, or execute code waiting for the transfer to complete. If additional bytes are required to complete the initial transfer, the DMA receive channel and the DMA transmit channel must be re-initialized and enabled. The process is repeated until the global count variable reaches zero.

Example Code

Example code for this application note is located on EDK web site http://developer.intel.com/design/edk/product/strongarm_edk.htm.