Module Introduction

Purpose
• This training module covers 68K/ColdFire Memory Architecture

Objectives
• Describe the cache configurations of V2, V3, and V4 ColdFire products.
• Explain the features of Embedded SRAM and the Embedded Flash Module.
• Explain the features of the EzPort serial flash programming interface.
• Describe the features of the Memory Controllers (DRAM and SDRAM).
• Explain the FlexBus muxing schemes available on MCF548x/7x devices.
• Describe the features of the Hardware Divide module and the Direct Memory Access (DMA) controller module.

Content
• 17 pages
• 3 questions

Learning Time
• 25 minutes

This module introduces you to the Coldfire Memory Architecture.

In this course, we will describe the cache configurations of V2, V3, and V4 ColdFire products, the embedded SRAM and the Embedded Flash Module including the EzPort serial flash programming interface. In addition to discussing the on-chip memories, this module discusses the features of the Memory Controllers (DRAM and SDRAM) and the FlexBus. We will also explore the Hardware Divide module and the Direct Memory Access (DMA) controller module.
Recent V2 ColdFire products use a configurable cache memory, V3 products use a unified cache memory, and V4 implements a Harvard architecture with separate data and instruction caches. However, in all versions, the cache is located on the core’s local bus, allowing it to service all fetches from the core in a single-cycle on cache hits. Also, all three versions support single-cycle access on cache hits, and the cache is implemented as a non-blocking design.

Select a button to learn more about each type of memory available in ColdFire products. After you have viewed all of the pages, click the forward arrow to advance to the next page.
Recent V2 ColdFire products use a configurable cache memory. The cache can be used for instructions only or data only. There is also an option to split the cache and use half of the cache for data and half for instructions.

The cache is a direct-mapped single-cycle memory. It consists of a tag array (containing addresses and a valid bit), and a data array.

There are two 16-byte line-fill buffers—one for instruction and one for data. The line-fill buffers provide temporary storage for the last line fetched in response to a cache miss. The core’s local bus is released after the initial access of a miss, allowing the cache or SRAM module to service additional requests while the remainder of the line from the cache miss is being fetched and loaded into the line-fill buffer.

The cache also supports a configurable miss fetch algorithm that will determine how data is loaded into the fill buffer based on the selected algorithm and the alignment of the access.

In addition, the V2 cache supports write-through cache mode.
Version 3 ColdFire products use a unified cache, which can be used for a mix of instructions and data.

The cache also uses a 4-way set-associative mapping instead of direct mapping as found on the V2.

The V3 has just one line-fill buffer, because it is a unified cache.

Unlike the V2 cache, the V3 cache has support for both write-through and copyback cache modes.
The V4 ColdFire processor implements a Harvard architecture with separate buses for instructions and data. Because different physical buses are used, two caches are required. The data cache is connected directly to the processor’s local data bus, and the instruction cache is located on the processor’s local instruction bus. The V4 also implements a branch instruction cache used to maximize the performance of conditional branch instructions. The branch cache implements instruction folding, which allows zero-cycle execution times for correctly predicted taken branches.

Both the instruction and data cache implement 16-byte line-fill buffers for each cache.

Same like V3 cache, the V4 cache has support for both write-through and copyback cache modes.
The SRAM module provides a general-purpose memory block that the ColdFire processor can access in a single cycle. Because the SRAM module is physically connected to the processor's high-speed local bus, it can service processor-initiated access or memory-referencing commands from the debug module.

The SRAM is dual-ported to provide access for the DMA or other on-chip masters.

Also, the SRAM is partitioned into two physical memory arrays to allow simultaneous access to both arrays by the processor core and another bus master.
Now, let's check your understanding of the cache features of various Coldfire products.

Correct.
The correct order for the answers are B, A, C.
ColdFire Flash Module (CFM)

Features
- The flash module itself can be up to 512 Kbytes of Flash memory
- Concurrent erase or blank verify of all flash array blocks
- Supports up to 80 MHz flash array read operations with 2-1-1-1 burst accesses
- Single power supply (Vdd, 3.3V) used for all module operations. No need for separate programming voltage
- Automated program and erase operation.
- Read-while-write capability on some devices
- 100,000 W/E cycles at room temperature and 10 years data retention
- Optional interrupt on command completion
- Protection scheme to prevent accidental program or erase
- Access restriction control for supervisor/user and data/program space operations
- Security for single-chip operations
- Auto sense amplifier timeout for low-power, low-frequency read operations

Let’s now turn our attention to the ColdFire Flash Module (CFM). This module is ideal for program and data storage for single-chip applications and allows for field reprogramming with no external high-voltage sources.

The Flash Module itself is 512 Kbytes and supports read, write, and program/erase operations. Bulk erase and page erase operations are supported. Page erase size depends on array size but is currently no larger than 2 Kbytes. Data is programmed in longword (32-bit) fashion.

Concurrent program and erase is supported on some devices.

Reads are executed as 2-1-1-1 burst accesses at up to 80 MHz.

The voltage required to program the CFM is generated internally by on-chip charge pumps.

Program and erase operations are automated.

It supports read while write capabilities.

Supports 100,000 W/E cycles at room temperature with data retention up to 10 years.

Optional interrupt on completion.

The CFM also contains a flexible protection scheme that will safeguard against accidental program and erase [operations] of the flash memory space to ensure that valuable data and code is not lost.

There is also access restriction control for supervisor/user and data/program space operations.

One of the main features of the CFM is its security operation. This feature prevents unauthorized user access to the CFM during single-chip operation. Security can be disabled via a special JTAG bulk erase command in the event of accidental lock out during system development.

Auto sense amplifier timeout for low-power, low-frequency read operations.
In CFM normal mode, the user can access the CFM registers and the CFM memory in order to perform operations such as read, program, erase, and blank check. The command sequence to perform flash operations is a simple three step process. The sequence is the same for erase verify, program, page erase, and mass erase operations. This allows for code sharing and shorter programs.

For reduced current draw, the device can enter stop mode. In this mode, the flash high voltage circuitry is switched off and any pending commands are aborted.

Flash can be protected by individual sectors. The protected sectors cannot be programmed or erased.

The CFM can also be secured to prevent unauthorized access to the module.

This secured mode disables the BDM port and forces the chip into single-chip mode to prevent boot from external memory. This secured mode can be bypassed in two ways. First, an 8 byte back door comparison key can be programmed at a particular memory address within the flash array. During the back door access, data is written to this same location. If the data written is identical to the data already residing at that location, security is unlocked. Second, security can be bypassed by bulk erasing the device using JTAG.
True or false? One of the main features of the ColdFire Flash Module (CFM) is its ability to prevent unauthorized user access to the CFM during single-chip operation. Click the correct answer and then click Done.

A) True
B) False

Please answer this question about the ColdFire Flash Module.

Correct!
One of the main features of the ColdFire Flash Module (CFM) is its ability to prevent unauthorized user access to the CFM during single-chip operation.
EzPORT is a serial flash programming interface that allows the flash memory contents on a 32 bit general purpose micro-controller to be read, erased, and programmed from off-chip in a compatible format to many stand-alone flash memory chips.

Once the on-chip flash has been programmed. The MCU can be reset so that the chip will boot from the on-chip Flash.

The EzPort can operate in one of two different modes: enabled and disabled.

When enabled, the EzPort ‘steals’ access to the Flash memory, preventing access from other cores or peripherals. The rest of the micro-controller should be disabled when the EzPort is enabled to avoid conflicts.

When the EzPort is disabled, the rest of the micro-controller can access Flash memory as normal.
Single Data Rate SDRAM Controller

- Supports up to 2 banks of DRAM
  - 5206e, 5307, and 5407 support ADRAM and SDRAM
  - 5282 and later support SDRAM only
- Supports 8-, 16-, & 32-bit wide DRAM banks
- Programmable wait states and refresh timer
- Support for page mode accesses
- Supports REFRESH operations
- Supports SELF REFRESH mode

The SDRAM controller provides control for /RAS, /CAS, and /DRAMW signals, as well as address multiplexing and bus cycle termination. On the 5206e, 5307, and 5407 the controller supports control signals and termination for both ADRAM or SDRAM. To reduce complexity, address line multiplexing is the same for both ADRAM and SDRAM operation. Starting with the 5282 support for ADRAM is eliminated and only SDRAM is supported.

The SDRAM controller module supports up to two banks of DRAM. These banks can be 8-, 16- or 32-bit wide DRAM. The module has programmable wait states and refresh timer. The SDRAM controller also outputs commands on the rising clock edge. During normal execution of application code any memory access to an external SDRAM will cause the correct sequence of commands to be performed to satisfy the read or write request.

The controller supports several different page mode options. The page mode options for ADRAM operation are different than those for SDRAM operation. When page mode is used the SDRAM controller can leave a page of the SDRAM open, meaning that the row address for that page does not need to be resent before accessing additional data.

The DRAM controller supports a REFRESH counter used to ensure that the SDRAM is refreshed periodically.

A SELF REFRESH command can be used to retain data in SDRAM memory even if the rest of the system is powered down. This is very valuable feature. When in self refresh mode the memory retains data without clocking. This means that SD_CLK can be stopped to reduce power consumption and EMI.
The MCF5275/74 products implement a 16-bit DDR SDRAM controller.

The controller provides a glueless interface to DDR1 SDRAM memories.

There are up to two chip selects available on the device, allowing for two separate blocks of memory.

Each block of memory can be up to 128 Mbytes, for a maximum of 256 Mbytes of memory.

The SDRAM controller uses a fixed 16-bit port width. Each block of SDRAM can be comprised of either a single 16-bit wide SDRAM chip or two 8-bit wide chips interfaced to the processor’s 16-bit bus.

Since DDR SDRAM clocks in data on both rising and falling clock edges, a 16-bit port allows for servicing of 32-bit accesses from the ColdFire core.

In order to help maximize overall memory throughput, the SDRAM controller supports page mode. During operation, the SDRAM controller maintains an open page address for each SD_CS block. The SDRAM controller also supports power-savings modes for the SDRAM such as self-refresh mode.
Some ColdFire products implement an SDR/DDR SDRAM controller. The controller provides a glueless interface to SDR and DDR1 SDRAM memories. Some implementations can also support mobile DDR. Refer to the device specific manual to determine which types of memories are supported.

The SDRAM controller supports programmable 32-bit or 16-bit port width. A 32-bit port width is recommended for SDR-based systems, and a 16-bit port size is recommended for most DDR designs. The MCF548x/547x implementation of the controller is a bit different than other devices. The 548x/7x devices use a fixed 32-bit port for the dedicated SDRAM controller bus and can address up to 4 blocks of SDRAM.

The maximum amount of memory that can be connected to the controller can vary from device to device. Most implementations of this controller can support up to 256 Mbytes of memory per chip select. Refer to the device reference manual to determine the supported memory configurations for the specific implementation.

The number of chip selects can also vary. The controller can support up to four chip selects, but the number of chip selects that are pinned out depend on the device.

Like the DDR only implementation of the controller, the DDR/SDR controller supports page mode operation and self-refresh low-power mode.
FlexBus – System Bus Controller

- Independent, user-programmable chip-select signals that can interface with SRAM, PROM, EPROM, EEPROM, Flash, or other peripherals
- 8-, 16-, and 32-bit port sizes
- Byte, word, longword, and line size transfers
- Programmable burst/burst-inhibited transfers selectable for each chip select and transfer direction
- Internal and external termination of bus cycles
  - Up to 63 wait states used for auto-acknowledge cycles
- New secondary wait state counter added for burst cycles
- Programmable address setup and hold time

The FlexBus module is used as the external system bus interface for ColdFire devices. It can be used to interface to Flash, SRAM, and other external memories or external peripherals that use a parallel bus. The FlexBus module controls the chip selects and control signal timing for external bus accesses. Most implementations support up to six FlexBus chip select signals. The number of chip selects available is device dependent.

The FlexBus supports 8, 16, and 32 bit port sizes and byte, word, longword, and line (16 byte) sized transfers to any port size.

For accesses that are larger than the port size, the chip select can be enabled to allow for bursting. With bursting disabled, any transfer that is larger than the port size is broken into multiple individual transfers. With bursting enabled, an access that is larger than the port size would result in a burst cycle of multiple beats. Bursting is programmable for each chip select and also each transfer direction. For example, a flash device might support burst accesses during reads, but not during program operations.

The FlexBus also supports two means of ending a bus cycle. An access can be externally terminated by asserting the transfer acknowledge (/TA) signal on the device. This signals to the CPU that the data has been latched by the external device for a write cycle or that data is ready to be latched by the CPU in the case of a read.

When interfacing to devices without a control signal to notify the processor that it can end the current access, internal termination can be used. When internal termination for a chip select is enabled a wait state counter is used to determine how many clocks the FlexBus should delay before asserting the /TA signal internally. The number of wait states to use can be programmed independently for each chip select.

The FlexBus also incorporates a secondary wait state counter that can be used for burst cycles. If the primary wait state counter is used, then the same number of wait states will be used for the initial access of a burst and all of the following beats. The secondary wait state counter allows for a different number of wait states for the initial access of a burst and the subsequent beats.

In addition the FlexBus has programmable options for address setup and hold time. Between 0 and 3 clocks of address setup and/or address hold time can be added to a bus cycle. These options are programmable on a per chip select basis. In the case of address hold, it is also programmable based on the transfer direction.
The MCF548x/7x devices use a multiplexed implementation of the FlexBus where a single 32-bit bus is used for both address and data. The full 32-bit address is always driven out at the beginning of the cycle. Then, during the data phase, some or all of the lines are used for data. The number of lines used for data is determined by the port size of the device.

For example, when interfacing to a 16-bit device the full 32-bit address is driven out on AD[31:0] during the address phase. Then during the data phase AD[31:16] are used for data, and the lower half of the bus (AD[15:0]) continues to drive the address for the transfer.

For most 32-bit devices a latch will be needed to capture the address for an access. However, for smaller port sizes a latch might not be necessary depending on the number of address lines needed.

The MCF548x/7x also supports a non-multiplexed FlexBus. At reset, non-muxed mode can be selected. In this mode the PCI bus is used to drive the address for FlexBus accesses, but the PCI module cannot be used in this mode.

### MCF548x/7x Muxing Schemes

|-----------|-----------|-----------|----------|---------|

- There is support for non-multiplexed 32-bit address and data using the PCI_AD[31:0] bus for FlexBus address.
Which of the following statements about the FlexBus module are true? Select all that apply and then click Done.

A. Between 0 and 4 clocks of address setup and/or address hold time can be added to a bus cycle.
B. It is used as the external system bus interface for ColdFire devices.
C. It incorporates a secondary wait state counter that can be used for burst cycles.
D. The number of wait states to use applies to all chip selects.

Done

Take a moment now to answer this question about the FlexBus module. Select all that apply and then click Done.

Correct!
B and C are true. The FlexBus module is used as the external system bus interface for ColdFire devices. Also, it incorporates a secondary wait state counter that can be used for burst cycles.
The Direct Memory Access (DMA) controller module provides a quick and efficient process for moving blocks of data with minimal processor overhead. The DMA module provides up to four independent channels that allow byte, word, longword or 128 bit block with bursting capability for data transfers. These transfers can be single or dual address to off-chip devices or dual address to on-chip devices.

The DMA transfer operation can be initiated internally or externally. It can be initiated internally by setting the Start bit in the DMA control register, by pulling one of the two external DMA request pins low, or by the internal UARTs. Single address transfers take one bus cycle. Dual address transfers take two bus cycles. Channel arbitration takes place on transfer boundaries.

Depending on your memory system this can be as fast as two bus clocks for single address transfers and four bus clocks for dual address transfers. Glue logic is normally not required for dual address applications but is almost always required for single address applications. The block size is determined by the 16-bit value in the byte count register which will allow up to 64 thousand bytes per block to be transferred.

The DMA has two Address Pointers: one for the source address and one for the destination address. The DMA supports auto alignment which means the accesses will be optimized based on the address value and the programmed size. The DMA supports data transfers to and from: memory to memory, peripheral to memory, and memory to peripheral.

Data packing and unpacking is also supported. This means that if the source and destination memory width is different, the bus accesses will accommodate the most efficient method to complete the transfer. For example, if you are reading from an 8-bit wide memory and writing to a 32-bit wide memory, the DMA will read the next four locations from the source memory and then make one write to the destination memory. This example assumes you are already memory aligned and you are not using burst memory. Source and destination pointer may be programmed to increment after transfer or not.

The DMA also supports continuous mode operation where a DMA cycle will run to completion after a single request and cycle steal operation where each DMA request will trigger a single read and a single write transfer.

On some ColdFires, the DMA has been enhanced to include channel linking, modulo addressing, and automatic disabling of external requests when the DMA completes.
The enhanced Direct Memory Access (eDMA) controller module provides an upgrade over the original ColdFire DMA unit. One of the main enhancements is that the eDMA offers 16, 32, or 64 channels as opposed to the original DMA’s four channels. The number of channels is implementation dependent.

The eDMA controller performs dual address transfers with programmable source and destination addresses. Flexible address incrementing is independently programmable for the source and destination address. The eDMA also implements modulo addressing features that can be very useful when accessing circular buffers.

The eDMA uses a 32 byte transfer controller descriptor to define the desired data movement. The transfer descriptors are accessible in the memory-mapped register space for the eDMA. Data movements are described using nested transfers. The inner (or minor) loop defines the byte transfer count for a single transfer, and the outer (or major) loop defines the number of times the inner loop executes. The combination of the inner and outer loop counters allows the eDMA to transfer large amounts of data without processor intervention.

The eDMA supports transfer sizes of 8 bits, 16 bits, 32 bits, and 16 bytes. The 16 byte internal data buffer allows for independent source and destination transfer sizes.

The arbitration between DMA channels is programmable. In round robin mode, the channels are cycled through without regard to priority. In fixed mode, each channel can be programmed with a priority level. In addition, the channel can optionally be programmed to allow preemption. This means that the channel will suspend operation and allow a higher priority channel to start.

The eDMA supports three different channel initiation methods. Software can start a DMA channel by setting the start bit for the appropriate channel. On-chip modules and external request signals can be used for hardware requests (the modules that can request DMA service and the number of external DMA request signals are implementation dependent). In addition, there is a channel linking feature that allows the completion of one DMA channel’s minor or major loop to signal a start to a second DMA channel.

There are also several programmable options for interrupt generation. Progress and error interrupts can be enabled on a per channel basis. Progress interrupts can be setup to interrupt the CPU when the DMA operation is completely finished or half done (that is, when half of the major loop iterations have been completed).

Finally, the eDMA supports a scatter/gather mode. If scatter/gather is enabled in a transfer control descriptor, when the channel completes the current TCD, the eDMA will automatically load the next TCD from memory. This allows the eDMA to automatically process a linked list of TCDs stored in memory without requiring service by the CPU.
In this module, you learned about the features of the on-chip memories including the cache, embedded SRAM, and embedded Flash.

You also learned about the EzPORT that can be used for serial programming of the embedded Flash.

And you learned about the features of the on-chip memory controllers that allow for glue-less interfacing to many types of external memory.

And finally Direct Memory Access controller offering up to four independent channels for quick and efficient process for moving blocks of data with minimal processor overhead.