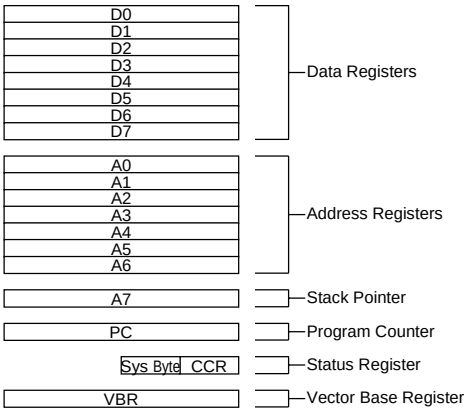
Module Introduction

- **PURPOSE:** The intent of this module, 68K to ColdFire® Transition, is to explain the changes to the programming model and architecture of ColdFire. This module also provides a description of the tools available to help with the transition from 68K to ColdFire.
- **OBJECTIVES:**
  - Describe the differences in the ColdFire programming module.
  - Identify the transition tools for migration from 68K to ColdFire.
- **CONTENT:**
  - 12 pages
  - 2 questions
- **LEARNING TIME:**
  - 25 minutes

The intent of this module is to explain the changes to the programming model and architecture of **ColdFire®** . This module also provides a description of the tools available to help with the transition from 68K to ColdFire. You will better understand the specific ColdFire devices as part of the upgrade path from 68K microprocessors.

Once you have finished this module, you will be better prepared to describe the differences between the 68K and ColdFire programming modules. You will also be better prepared to identify the available transition tools that help simplify the migration process from 68K to ColdFire.

ColdFire Programming Model

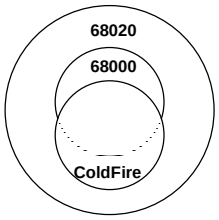


Let's begin with a look at the ColdFire programming model. Does it look familiar? This may be because it is nearly identical to that of the M68K family.

There is one main difference between the stack pointer of this programming model and the M68K family. That difference is, there is only one stack pointer in the ColdFire architecture. Keep in mind that the M68K supported a user and a supervisor stack pointer. This provided operating system support not required on ColdFire. Note that all ColdFires except for the MCF5282 only implement[s] a single stack pointer (A7). Please refer to the ColdFire Programmer's Reference Manual for more detail.

Next, let's look at the ColdFire instruction set architecture.

Instruction Set Architecture



- **Deleted from M68000**
- **Included in ColdFire**
- **Added beyond M68000**

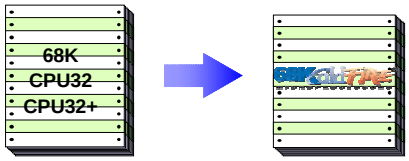
Next, let's look at the ColdFire instruction set architecture. Many instructions were eliminated from the M68K. Many of these instructions were rarely used or could be replaced by a sequence of simpler instructions, as in the reduced instruction set computing (RISC) tradition. The instructions that were deleted from M68000 include BCD instructions, rotate instructions, simplified fault model, double stack pointer, and eliminated support for read-modify-write (r-m-w) instructions. Also, reduced support for byte and word instructions means that only double word operands, such as 32-bit, are supported in memory. With the introduction of the Version 4 core, many of these operand size restrictions have been eliminated.

Included in ColdFire are full support for basic effective addressing modes; full support for move byte, word, and long operands; and supervisor and user privilege modes. Note that the full complement of byte, word, and long word instructions are supported in the internal register set.

Added beyond M68000 are longword multiplies; extend byte to long instruction; trapf, trapf.w and trapf.l; BDM instructions; and vector base register. Also note the inclusion of x2 and x4 scaling factors. This is helpful in the manipulation of data arrays. Additional background debug capability has also been added.

Assembler Translator

- Migration from 68K to ColdFire
- Free MicroAPL PortASM/68K software
- Tool features
- Tool and support availability



ColdFire Variable-Length RISC architecture is derived from the 680x0 family, but with a simplified set of instructions. It also addresses modes to provide high performance at low-cost and low-power consumption. This means that programs written for the 680x0 family will not run on ColdFire processors without modification or re-compilation.

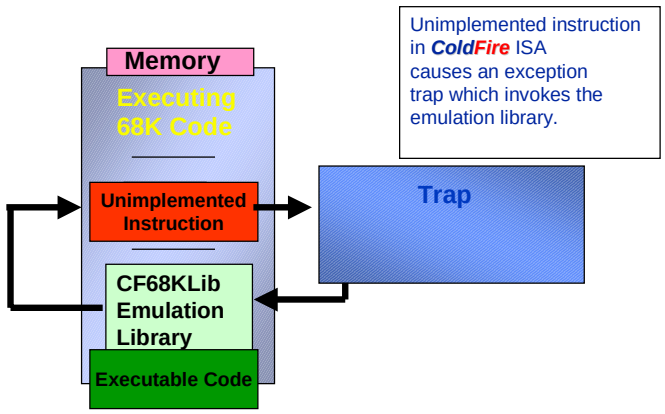
The first solution is PortASM/68K Software for ColdFire. This free solution is a source-level translation tool that is designed to help you port 680x0 assembly-language programs to ColdFire. This automated tool will translate 680x0, CPU32, and CPU32+ assembly source code to ColdFire family source code. It can be used to target any member of the ColdFire family. It provides good performance and is the recommended solution for porting a large body of 680x0 assembler code where you have access to the original source-code files.

There are many features of the PortASM/68K tool. It can choose between optimization of code size or performance when doing translation. Also, it can choose between ColdFire core versions, with or without the media access controller (MAC), and divide unit. Another important feature of this tool is that the program flags, with recommended corrections, 68K code that PortASM/68K cannot readily convert. This tool also allows debug instructions to be inserted into code. This gives users the flexibility to use tools that take advantage of the ColdFire debug module.

PortAsm/68K Software can be downloaded free of charge from the web site (http://www.Freescale.com). Free installation support is available and full support is available from MicroAPL for less than $500.

Now, let’s look at the next free solution for migrating from 68K to ColdFire.

68K Emulation Library



Summary of Events:

1. **Execute 68K code on *ColdFire* processor**
2. **Trap on illegal instruction**
3. **Emulate instruction**
4. **Return to execute code**

The next available solution is CF68KLib. This is an emulation library that provides exception handlers to implement 680x0 instructions. It also addresses modes that are missing in the ColdFire architecture. It allows 680x0 object-code, whether written in assembler or in a high-level language, to run on a ColdFire processor.This includes all ColdFire microprocessors, with the exception of the MCF5206 and MCF5206E. It is available to run either user-level code only, or user- and supervisor-level code. CF68KLib allows you to run 680x0 code with minimal modification under ColdFire. However, there is a performance penalty associated with the need to trap out and emulate missing instructions.

The CF68Lib emulation library supports the following 68K processors: 68000, 68020, 68030, 68EC040, 68EC060, CPU32, and CPU32+. However, it does not support floating point or memory management unit (MMU) instructions. Not only is this tool an excellent resource for protecting legacy 68K code developed by our customers, but it is offered free of charge. It is available for ColdFire processors with full instruction decode. CF68Lib is supported on Win95/98, SunOS, Solaris, and WinNT.

For additional information about the available solutions to ease the migration from 68K to ColdFire, visit the web site (http://www.freescale.com)

Translation Questions

- **All of our design expertise hinges on 68K devices. Would this expertise be lost when moving to ColdFire?**
- **ColdFire will operate at a much higher frequency than 68K. Can I still use existing peripherals that are much slower ?**
- **A large amount of initialization code exists for 68K architecture. Will this code have to be re-written?**

Here are some frequently-asked questions that arise from the 68K to ColdFire transition.

All of our design expertise hinges on 68K devices. Would this expertise be lost when moving to ColdFire? The answer to this question is that the ColdFire instruction set is a subset of the 68K instruction set. Also, ColdFire code is compatible at both the binary and assembler level with original 68K instructions. Programming models are identical with the exception of the stack pointer implementation. The exception processing model in ColdFire is also similar to the original 68K instructions.

ColdFire will operate at a much higher frequency than 68K. Can I still use existing peripherals that are much slower? The answer to this question is that it is still possible to use slower peripherals by inserting wait states and glueless interface. Using existing peripherals may adversely affect performance. To increase performance, upgrade the peripherals.

A large amount of initialization code exists for 68K architecture. Will this code have to be re-written? The answer to this question is that initialization code is specific to a particular device. Normally new code would have to be written, but this initialization code is provided on the web site (www.freescale.com). Many of the peripheral functions found on 68K devices have been implemented on ColdFire devices. Much of the initialization code for a 68K device may be re-used on ColdFire devices with very little modification.

Question

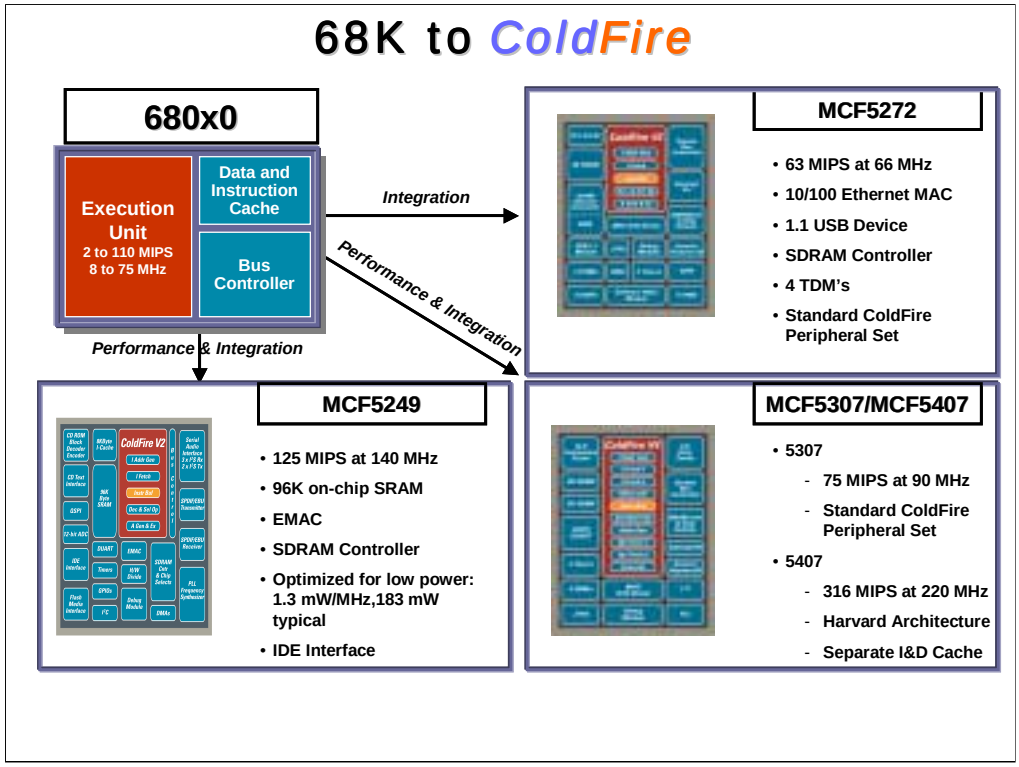
Which of the following does ColdFire include?
Click on your choice and then click Done.

a) Full support for more byte, word, and long operands.
b) BCD instructions.
c) Rotate instructions.
d) Divide instructions.

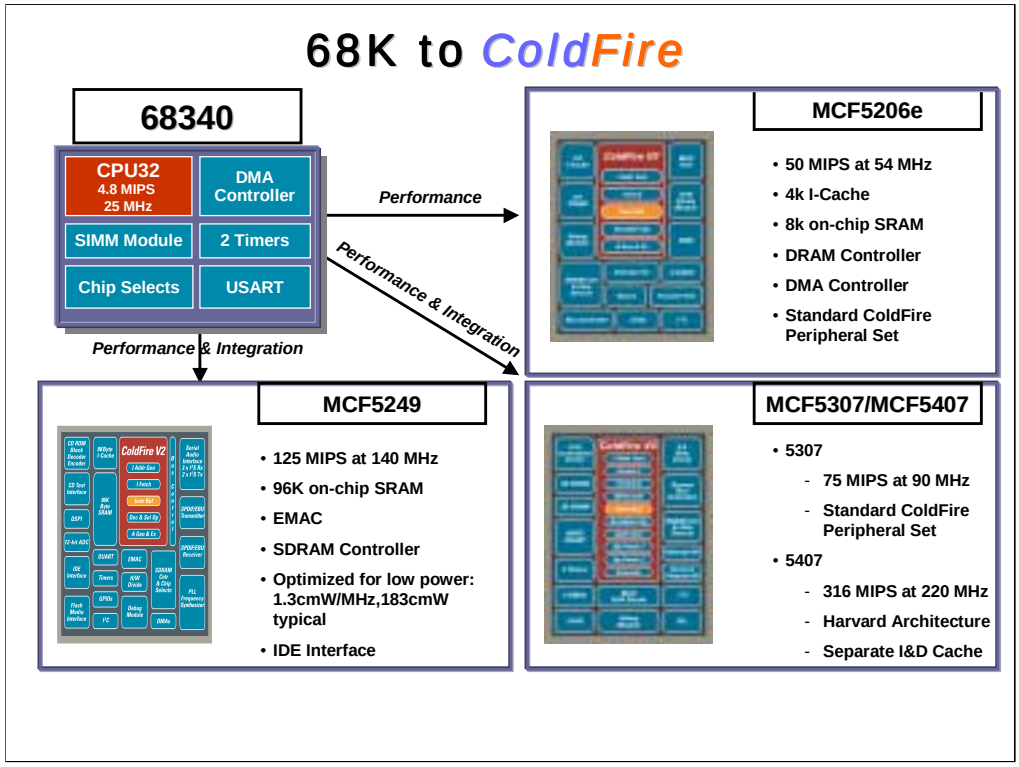
Here's another question for you.

Answer:

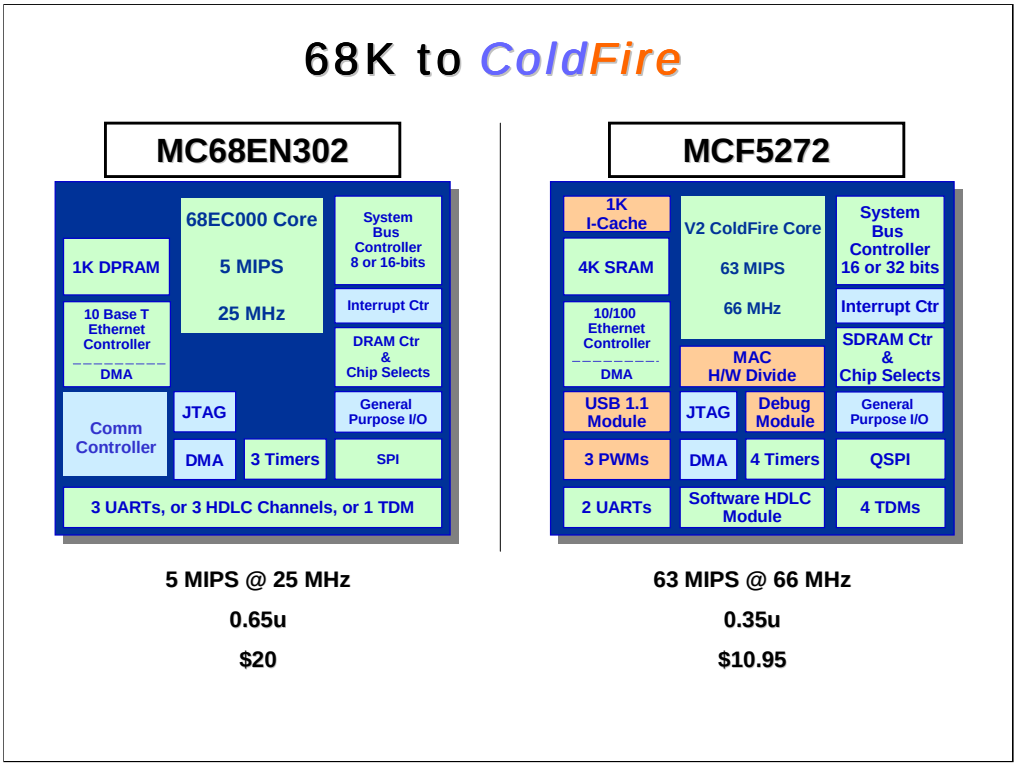
ColdFire includes full support for more byte, word, and long operands. BCD, rotate, and [divide] read-modify-write instructions have all been deleted.



Here are some recommendations showing which ColdFire devices best fit as a migration from 680x0 microprocessors. All of these migration paths offer more integration and performance typically at a lower price point.



These are the recommended ColdFire devices that best fit as a migration from the popular 68340 microprocessors. Once more, all of these migration paths offer more integration and performance.



Some of our 68K microprocessors are application specific, as in the case of the 68302 family. This family, especially the MC68EN302, is well renowned for its networking peripheral set. In the case of the MC68EN302, there is an onboard 10-Base-T Ethernet controller. Therefore, the MCF5272 ColdFire is the best possible replacement due to the fact that the MCF5272 features a 10/100 Ethernet controller. To help understand this page, the two block diagrams are color-coded to represent the enhancements of the MCF5272 from the MC68EN302.

These highlighted boxes indicate a feature or peripheral that is enhanced on the MCF5272.

These highlighted boxes indicate features or peripherals that are brand new on the MCF5272.

These highlighted boxes indicate features and peripherals that are matched on the MCF5272. In addition, the MCF5272 has over 12x the performance for only half the price of an MC68EN302! Migrating from 68K to ColdFire offers higher performance, newer technology, and a lower price.

Question

Which tools are available to assist in the migration from 68K to ColdFire? Select all that apply and then click Done.

CF68KLib: an emulation library that provides exception handlers to implement 680x0 instructions.

PortASM/Lib: an emulation library that allows you to run 680x0 code with minimal modification.

CF68Port: a migration solution that can be downloaded free of charge.

PortASM/68K Software: a source-level translation tool that is designed to help you port 680x0 assembly-language programs to ColdFire.

Consider this question about the migration from 68K to ColdFire.

Answer:

PortASM/68K Software is a free solution that can be used to target any member of the ColdFire family. CF68KLib is an emulation library that also addresses modes that are missing in the ColdFire architecture.

Module Summary

- ColdFire programming model
- Instruction set architecture
- MicroAPL PortASM/68K software
- 68K emulation library
- Translation questions
- Recommended ColdFire devices

Let’s review the 68K to ColdFire Transition module. First, we examined the ColdFire programming model. Then, we looked at the type of instructions that were removed, included, and added to ColdFire. Next, we looked at two solutions to help ease the migration from 68K to ColdFire. We answered some frequently-asked questions about the 68K to ColdFire transition. Finally, we presented recommendations showing which ColdFire devices best fit as a migration from 680x0 microprocessors.